

abort Method (DOMDocument)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

Applies to

See Also

Aborts an asynchronous download in progress.

JScript Syntax

 Copy

```
oXMLDOMDocument.abort();
```

Example

jscript

 Copy

```
var xmlDoc;
xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
xmlDoc.async = true;
xmlDoc.onreadystatechange = doOnReadyStateChange;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
}

function doOnReadyStateChange () {
    if (xmlDoc.readyState == 1) {
        xmlDoc.abort();
    }
}
```

C/C++ Syntax

```
HRESULT abort(
    void
);
```

 Copy

Return Values

S_OK

The value returned if successful.

Remarks

This method stops download and parsing and discards any portion of the XML tree already built. `IXMLDOMParseError` indicates that the download was stopped.

If the `readyState` property has the value `COMPLETED`, no action is taken and the current document is unchanged.

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[IXMLDOMParseError](#)

[readyState Property \(DOMDocument\)](#)

abort Method (IXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Visual Basic Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Cancels the current HTTP request.

JScript Syntax

 Copy

```
oXMLHttpRequest.abort();
```

Example

jscript

 Copy

```
<SCRIPT ID=clientEventHandlersJS LANGUAGE=javascript>
<!--
  function XMLHTTPAbort_onclick() {
    var xmlhttp = new ActiveXObject("MSXML2.XMLHTTP.6.0");
    xmlhttp.onreadystatechange = doHttpReadyStateChange;
    xmlhttp.open("GET",
"http://<%=Request.ServerVariables("Server_Name")%>/sample.xml", true);
    xmlhttp.send();
    // alert(xmlhttp.responseXML.xml);

    function doHttpReadyStateChange() {
      switch (xmlhttp.readyState) {
```

```
// The following lines allow completion of the XMLHTTP request  
// case 4: alert(xmlhttp.responseXML.xml);  
// default: return;  
  
// The following lines abort the XMLHTTP request  
case 2: xmlhttp.abort();  
default: alert(xmlhttp.responseXML.xml);  
}  
}  
  
}  
//>  
</SCRIPT>  
<INPUT type="button" value="Abort XMLHTTP" id=XMLHTTPAbort name=XMLHTTPAbort  
LANGUAGE=javascript onclick="return XMLHTTPAbort_onclick()">
```

Visual Basic Syntax

 Copy

```
oXMLHttpRequest.abort
```

C/C++ Syntax

 Copy

```
HRESULT abort(void);
```

Return Values

S_OK

The value returned if successful.

Remarks

The request will be returned to the UNINITIALIZED state, and `open` method must be called next.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLHTTPRequest](#)

See Also

[readyState Property \(IXMLHTTPRequest\)](#)

[open Method \(IXMLHTTPRequest\)](#)

abort Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Example](#)

[C/C++ Sntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Cancels the current HTTP request.

JScript Syntax

 Copy

```
oServerXMLHttpRequest.abort();
```

Example

JavaScript

 Copy

```
var xmlServerHttp;  
xmlServerHttp = new ActiveXObject("Msxml2.ServerXMLHTTP.3.0");  
xmlServerHttp.onreadystatechange = doServerHttpReadyStateChange;  
xmlServerHttp.open("GET", "http://localhost/sample.xml", true);  
xmlServerHttp.send();  
  
function doServerHttpReadyStateChange() {  
    if (xmlServerHttp.readyState == 2) {  
        xmlServerHttp.abort();
```

```
}
```

C/C++ Syntax

 Copy

```
HRESULT abort(void);
```

C/C++ Return Values

S_OK

The value returned if successful.

Remarks

The object will be returned to the UNINITIALIZED state, and `open` method must be called next.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IServerXMLHTTPRequest](#)-[ServerXMLHTTP](#)

See Also

[readyState Property \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[open Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

Abort Method (IXMLHTTPRequest2, Windows 8)

10/27/2016 • 2 minutes to read

In this article

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Cancels the current HTTP request.

C/C++ Syntax

 Copy

```
HRESULT abort(void);
```

C/C++ Return Values

S_OK

The value returned if successful.

Remarks

The object will be returned to the UNINITIALIZED state, and `open` method must be called next.

Versioning

Implemented in: MSXML 3.0 and later

Applies to

[IXMLHttpRequest2 \(Windows 8\)](#)

See Also

[readyState Property \(ServerXMLHTTP-IIServerXMLHttpRequest\)](#)

[IXMLHttpRequest2 \(Windows 8\)](#)

add Method (IXMLDOMSchemaCollection- XMLSchemaCache)

05/17/2017 • 4 minutes to read

In this article

[JScript Syntax](#)

[Example](#)

[C/C++ Syntax](#)

[Resource Files](#)

[Output](#)

[Remarks for MSXML 6.0](#)

[Versioning](#)

[Applies to](#)

Adds a new schema to the schema collection and associates the given namespace URI with the specified schema.

JScript Syntax

 Copy

```
objXMLDOMSchemaCol.add(namespaceURI, var);
```

Parameters

`namespaceURI`

The namespace to associate with the specified schema. The empty string, "", will associate the schema with the empty namespace, `|`.

This may be any string that can be used in an `xmlns` attribute, but it cannot contain entity references. The same white space normalization that occurs on the `xmlns` attribute also

occurs on `namespaceURI` (that is, leading and trailing white space is trimmed, new lines are converted to spaces, and multiple adjacent white space characters are collapsed into one space).

`var`

This specifies the schema to load. It will load it synchronously and with `resolveExternals=false` and `validateOnParse=false`. This parameter can also take any `DOMDocument` as an argument.

This argument can be Null, which results in the removal of any schema for the specified namespaces. If the schema is an `IXMLDOMNode`, the entire document the node belongs to will be preserved.

Example

The following script example attaches a schema to an XML document.

```
jscript Copy

var xmldoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var SchemaCache = new ActiveXObject("Msxml2.XMLSchemaCache.6.0");

xmldoc.async = false;
xmldoc.validateOnParse = false;
SchemaCache.add("x-schema:books", "collection.xdr");
xmldoc.schemas = SchemaCache;
// The document will load only if a valid schema is attached to the xml
// file.
xmldoc.load("collection.xml");
if (xmldoc.parseError.errorCode != 0) {
    var myErr = xmldoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    WScript.Echo(xmldoc.xml) ;
}
```

C/C++ Syntax

```
Copy

HRESULT add(BSTR namespaceURI, VARIANT var);
```

Parameters

`namespaceURI` [in]

The namespace to associate with the specified schema.

The empty string, "", will associate the schema with the empty namespace, `.`. This may be

any string that can be used in an `xmlns` attribute, but it cannot contain entity references.

The same white space normalization that occurs on the `xmlns` attribute also occurs on this parameter (that is, leading and trailing white space is trimmed, new lines are converted to spaces, and multiple adjacent white space characters are collapsed into one space).

`var` [in]

This specifies the schema. It can be a BSTR, in which case it points to the URL to load. It will load it synchronously and with `resolveExternals=false` and `validateOnParse=false`. The

`var` parameter can also be any `DOMDocument`.

This argument can be Null, which results in the removal of any schema for the specified namespaces. If the schema is an `IXMLDOMNode`, the entire document the node belongs to will be preserved.

Return Values

If this call fails, the collection remains unchanged. `E_FAIL` is returned if:

- The collection is read-only.
- The document is not a recognized schema.
- An error occurs when compiling the schema.
- The ready state of the document is not 4.

If it was loading a schema and encountered a parse error, then the parse error reason is returned in the `IErrorInfo`. If the VARIANT argument contains an invalid value, `E_INVALIDARG` is returned.

Example

C++

 Copy

```
#include "stdafx.h"
#include "tchar.h"
#include "msxml6.h"

void AddCollectionSample();

int APIENTRY WinMain(HINSTANCE hInstance,
                     HINSTANCE hPrevInstance,
                     LPSTR     lpCmdLine,
                     int       nCmdShow)
{
    ::CoInitialize(NULL);
    AddCollectionSample();
    ::CoUninitialize();
    return 0;
}

void AddCollectionSample()
{
    IXMLDOMDocument2Ptr pIXMLDOMDocument2;
    IXMLDOMSchemaCollection2Ptr pIXMLDOMSchemaCollection2Ptr;
    int nResult;

    try
    {
        // Create the DOM
        nResult =
pIXMLDOMDocument2.CreateInstance(__uuidof(MSXML2::DOMDocument60));
        (nResult == 0) ? 0: throw nResult;

        // Create the Schema Collections
        nResult =
pIXMLDOMSchemaCollection2Ptr.CreateInstance(__uuidof(MSXML2::XMLSchemasCache60))
;
        (nResult == 0) ? 0: throw nResult;

        // Add the schema to the collection
        nResult = pIXMLDOMSchemaCollection2Ptr->add(_T("x-schema:books"),
_T("c:\\temp\\collection.xsd"));
        (nResult == 0) ? 0: throw nResult;

        // Attach schemas
        pIXMLDOMDocument2->schemas =
pIXMLDOMSchemaCollection2Ptr.GetInterfacePtr();

        pIXMLDOMDocument2->async = false;
        pIXMLDOMDocument2->validateOnParse = true;

        // Load the document into the DOM
        nResult = pIXMLDOMDocument2->load(_T("c:\\temp\\collection.xml"));
        (nResult == -1) ? 0: throw nResult;

        ::MessageBox(NULL, pIXMLDOMDocument2->xml, _T("Loaded Document"),

```

```
    MB_OK);
} catch(...)

{
    ::MessageBox(NULL, _T("Sample Failed"), _T("Error"), MB_OK);
}
}
```

Resource Files

The examples in this topic use the following files.

collection.xml

```
<?xml version='1.0'?>
<Collection xmlns="x-schema:books">
    <Book>
        <Title>Lover Birds</Title>
        <Author>Cynthia Randall</Author>
        <Publisher>Lucerne Publishing</Publisher>
    </Book>
</Collection>
```

 Copy

collection.xsd (MSXML 4.0 and later)

```
<?xml version="1.0" encoding="UTF-8"?>
<xss:schema xmlns:xss="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
    <xss:element name="Collection">
        <xss:complexType>
            <xss:sequence>
                <xss:element name="OtherBook">
                    <xss:complexType>
                        <xss:sequence>
                            <xss:element name="Title" type="xs:string"/>
                            <xss:element name="Author" type="xs:string"/>
                            <xss:element name="Publisher" type="xs:string"/>
                        </xss:sequence>
                    </xss:complexType>
                </xss:element>
            </xss:sequence>
        </xss:complexType>
    </xss:element>
</xss:schema>
```

 Copy

collection.xdr (MSXML 3.0 and later)

```
XML Copy  
  
<?xml version="1.0"?>  
<Schema xmlns="urn:schemas-microsoft-com:xml-data">  
  <!-- AttributeType name="xmlns" -->  
  <ElementType name="Title"/>  
  <ElementType name="Author"/>  
  <ElementType name="Publisher"/>  
  <ElementType name="Book" model="closed">  
    <element type="Title"/>  
    <element type="Author"/>  
    <element type="Publisher"/>  
  </ElementType>  
  <ElementType name="Collection" model="closed">  
    <element type="Book"/>  
  </ElementType>  
</Schema>
```

Output

```
<?xml version='1.0'?>  
  
<Collection xmlns="x-schema:books">  
  
<Book>  
  
<Title>Lover Birds</Title>  
  
<Author>Cynthia Randall</Author>  
  
<Publisher>Lucerne Publishing</Publisher>  
  
</Book>  
  
</Collection>
```

Remarks for MSXML 6.0

In previous versions of MSXML, a schema already in the cache for a given namespace was replaced by the schema from the new location. In MSXML 6.0, when you call this method the declarations are merged with an existing schema with the same namespace. Using this

feature, MSXML 6.0 supports "partial schemas". You can load several schemas, all having the same target namespace, into one schema in the schema cache.

Calling this method will cause all the schemas imported by the added schema to also be added into the cache as "top-level" schemas.

Schema imports are validated "lax" - this means that any namespace or type already added to the schema cache can be referenced by another schema in the cache, even if there is no explicit import in the referencing schema. You need to set [validateOnLoad Property](#) to `false` to avoid issues around the order of calls to this method.

The add operation is atomic. All the schemas must be added successfully to the cache, or else none are.

MSXML 6.0 has removed support for XDR schemas, whereas XDR is supported in MSXML 3.0 AND MSXML 4.0. If this method is called with an XDR schema, the call will fail.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMSchemaCollection-XMLSchemaCache](#)

addCollection Method

10/27/2016 • 3 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Resource Files](#)

[Output](#)

[Remarks](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[This feature was implemented only for MSXML 6.0.]

Adds schemas from another collection into the current collection and replaces any schemas that collide on the same namespace URI.

JScript Syntax

 Copy

```
objXMLDOMSchemaCol.addCollection(objXMLDOMSchemaCollection);
```

Parameters

`objXMLDOMSchemaCollection`

The collection containing the schemas to add.

Example

```
jscript
```

 Copy

```
var xmldoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var SchemaCache = new ActiveXObject("Msxml2.XMLSchemaCache.6.0");
var SchemaCache2 = new ActiveXObject("Msxml2.XMLSchemaCache.6.0");

xmldoc.async = false;
xmldoc.validateOnParse = false;
SchemaCache.add("x-schema:books", "collection.xsd");
SchemaCache2.addCollection(SchemaCache);
SchemaCache2.add("x-schema:books", "NewBooks.xsd");
xmldoc.schemas = SchemaCache2;
// The document will load only if a valid schema is attached to the xml
// file. The new schema will override the old one
xmldoc.load("collection.xml");
if (xmldoc.parseError.errorCode != 0) {
    var myErr = xmldoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    WScript.Echo(xmldoc.xml) ;
}
```

C/C++ Syntax

 Copy

```
HRESULT addCollection(IXMLDOMSchemaCollection * otherCollection);
```

Parameters

otherCollection[in]

The collection containing the schemas to add.

C/C++ Return Values

S_OK

The value returned if the collection is added successfully.

E_FAIL is returned if:

- The collection is read-only.
- The document is not a recognized schema.
- An error occurs when compiling the schema.

- The ready state of the document is not 4.

E_POINTER

The value returned if input pointer is invalid.

Example

```

C++ Copy

#include "msxml6.h"

void AddCollectionSample();

int APIENTRY _tWinMain(HINSTANCE hInstance,
                      HINSTANCE hPrevInstance,
                      LPTSTR lpCmdLine,
                      int nCmdShow)
{
    if(SUCCEEDED(::CoInitialize(NULL))) {
        AddCollectionSample();
        ::CoUninitialize();
    }
    return 0;
}

void AddCollectionSample()
{
    IXMLDOMDocument2Ptr pIXMLDOMDocument2;
    IXMLDOMSchemaCollection2Ptr pIXMLDOMSchemaCollection2Ptr;
    IXMLDOMSchemaCollection2Ptr pIXMLDOMSchemaCollection2Ptr2;
    int nResult;

    try
    {
        // Create the DOM
        nResult =
pIXMLDOMDocument2.CreateInstance(__uuidof(MSXML2::DOMDocument60));
        (nResult == 0) ? 0: throw nResult;

        // Create the Schema Collections
        nResult =
pIXMLDOMSchemaCollection2Ptr.CreateInstance(__uuidof(MSXML2::XMLSchemasCache60))
;
        (nResult == 0) ? 0: throw nResult;
        nResult =
pIXMLDOMSchemaCollection2Ptr2.CreateInstance(__uuidof(MSXML2::XMLSchemasCache60)
);
        (nResult == 0) ? 0: throw nResult;

        // Add the schema to the collection
        nResult = pIXMLDOMSchemaCollection2Ptr->add(_T("x-schema:books"),

```

```

_T("collection.xsd"));
    (nResult == 0) ? 0: throw nResult;

    // attach all schemas from "pIXMLDOMSchemaCollection2Ptr"
    nResult = pIXMLDOMSchemaCollection2Ptr2-
>addCollection(pIXMLDOMSchemaCollection2Ptr.GetInterfacePtr());
    (nResult == 0) ? 0: throw nResult;

    // override the old schema
    nResult = pIXMLDOMSchemaCollection2Ptr2->add(_T("x-schema:books"),
_variant_t(_T("Newbooks.xsd")));
    (nResult == 0) ? 0: throw nResult;

    // Attach schemas
    pIXMLDOMDocument2->schemas =
pIXMLDOMSchemaCollection2Ptr2.GetInterfacePtr();

    pIXMLDOMDocument2->async = false;
    pIXMLDOMDocument2->validateOnParse = true;

    // Load the document into the DOM
    nResult = pIXMLDOMDocument2->load(_T("collection.xml"));
    (nResult == -1) ? 0: throw nResult;

    ::MessageBox(NULL, pIXMLDOMDocument2->xml, _T("Loaded Document"),
MB_OK);
} catch(...)
{
    ::MessageBox(NULL, _T("Sample Failed"), _T("Error"), MB_OK);
}
}

```

Resource Files

The examples in this topic use the following files.

collection.xml

```
<?xml version='1.0'?>
<Collection xmlns="x-schema:books">
    <Book>
        <Title>Lover Birds</Title>
        <Author>Cynthia Randall</Author>
        <Publisher>Lucerne Publishing</Publisher>
    </Book>
</Collection>
```

Copy

collection.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xss:schema xmlns:xss="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xss:element name="Collection1">
    <xss:complexType>
      <xss:sequence>
        <xss:element name="OtherBook">
          <xss:complexType>
            <xss:sequence>
              <xss:element name="Title" type="xss:string"/>
              <xss:element name="Author" type="xss:string"/>
              <xss:element name="Publisher" type="xss:string"/>
            </xss:sequence>
          </xss:complexType>
        </xss:element>
      </xss:sequence>
    </xss:complexType>
  </xss:element>
</xss:schema>
```

 Copy

NewBooks.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xss:schema xmlns:xss="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
  <xss:element name="Collection">
    <xss:complexType>
      <xss:sequence>
        <xss:element name="Book">
          <xss:complexType>
            <xss:sequence>
              <xss:element name="Title" type="xss:string"/>
              <xss:element name="Author" type="xss:string"/>
              <xss:element name="Publisher" type="xss:string"/>
            </xss:sequence>
          </xss:complexType>
        </xss:element>
      </xss:sequence>
    </xss:complexType>
  </xss:element>
</xss:schema>
```

 Copy

Output

The examples in this topic output the following in a message box.

 Copy

```
<?xml version='1.0'?>
<Collection xmlns="x-schema:books">
  <Book>
    <Title>Lover Birds</Title>
    <Author>Cynthia Randall</Author>
    <Publisher>Lucerne Publishing</Publisher>
  </Book>
</Collection>
```

Remarks

There is no guarantee that the two collections will return exactly the same schema. The collection is free to clone them, if necessary.

Adding a collection to itself has no effect.

Remarks

In previous versions of MSXML, a schema already in the cache for a specified namespace was replaced by the schema from the new location. In MSXML6.0, when you call the `addCollection` method the declarations are merged with an existing schema with the same namespace. Using this feature, MSXML 6.0 supports "partial schemas". You can load several schemas, all with the same target namespace, into one schema in the schema cache.

Calling this method will cause all the schemas imported by the added schemas to also be added into the cache as "top-level" schemas.

Schema imports are considered to be "lax" - this means that any namespace or type already added to the schema cache can be referenced by another schema in the cache even if there is no explicit import in the referencing schema. You must set [validateOnLoad Property](#) to false to avoid issues around the order of calls to this method.

The add operation is atomic. All the schemas must be added successfully to the cache, or else none are.

MSXML 6.0 has removed support for XDR schemas, whereas XDR is supported in MSXML 3.0. If this method is called with an XDR schema, the call will fail.

Versioning

Implemented in: MSXML 6.0

Applies to

[IXMLDOMSchemaCollection-XMLSchemaCache](#)

addObject Method (IXSLProcessor)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource File](#)

[Output](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

Adds objects into a style sheet.

JScript Syntax

 Copy

```
objXSLProcessor.addObject(obj, namespaceURI);
```

Parameters

obj

The object to pass in. Optionally, you can pass a null value here to signal to the processor that an object previously added should be released.

namespaceURI

The namespace to use inside the style sheet to identify the object.

Example

The following script example passes an object to the style sheet.

```
jscript
```

 Copy

```

var xsldoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var xmldoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var xsltemp = new ActiveXObject("Msxml2.XSLTemplate.6.0");
var xslproc;

xsldoc.load("sampleXSLWithObject.xml");
if (xsldoc.parseError.errorCode != 0) {
    var myErr = xsldoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xsltemp.stylesheet = xsldoc.documentElement;
    xslproc = xsltemp.createProcessor();
    xmldoc.loadXML("<level>Twelve</level>");
    xslproc.input = xmldoc;

    xslproc.addObject(xmldoc, "urn:my-object");
    xslproc.transform();
    WScript.Echo(xslproc.output);
}

```

Resource File

The JScript example uses the following file.

sampleXSLWithObject.xml

<pre> <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0" xmlns:myObj="urn:my-object"> <xsl:output method="xml" indent="yes"/> <xsl:template match="/"> <xsl:element name="stage"> <xsl:value-of select="myObj:get-text()"/> </xsl:element> </xsl:template> </xsl:stylesheet> </pre>	Copy
---	---

Output

XML	Copy
<pre> <?xml version="1.0" encoding="UTF-16"?> <stage>Twelve</stage> </pre>	

C/C++ Syntax

Parameters

obj[in]

The object to pass in. Optionally, you can pass a NULL IDispatch here to signal to the processor that an object previously added should be released.

namespaceURI[in]

The namespace that will be used inside the style sheet to identify the object.

Return Values

E_FAIL if the value of the `readyState` property is `READYSTATE_INTERACTIVE`.

Remarks

Numbers are coerced to double, everything is coerced into a string, and objects return an error.

The syntax `get-` is used to retrieve the property value exposed by the object that was passed into the style sheet. In the preceding example, the value for the property '`'text'`' was retrieved as follows:

```
<xsl:value-of select="myObj:get-text()"/>
```

For example, if the object that was passed into the style sheet included a property called `sheepcount`, you would use the following syntax to retrieve the value of that property.

```
<xsl:value-of select="myObj:get-sheepcount()"/>
```

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXSLProcessor](#)

addParameter Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Example](#)

[C/C++ Syntax](#)

[Resource File](#)

[Output](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

Adds parameters into an XSL Transformations (XSLT) style sheet.

JScript Syntax

 Copy

```
objXSLProcessor.addParameter(baseName, parameter, namespaceURI);
```

Parameters

`baseName`

The name that will be used inside the style sheet to identify the parameter context.

`parameter`

In most cases, a number, Boolean, string, `IXMLDOMNodeList`, or `IXMLDOMNode`. Passing in a single node will produce a node list that contains one node (shortcut). For MSXML 6.0, to remove a parameter previously added to the processor, provide a value of Empty or Null instead. This acts as a signal to the processor to remove any previously added parameter of the same name. However for MSXML3.0, this function cannot remove previously added parameters.

`namespaceURI` (optional)

An optional namespace.

Example

jscript

 Copy

```
var xslt = new ActiveXObject("Msxml2.XSLTemplate.6.0");
var xsldoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var xslproc;
xsldoc.async = false;
xsldoc.load("sample.xsl");
if (xsldoc.parseError.errorCode != 0) {
    var myErr = xsldoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xslt.stylesheet = xsldoc;
    var xmldoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
    xmldoc.async = false;
    xmldoc.load("books.xml");
    if (xmldoc.parseError.errorCode != 0) {
        var myErr = xmldoc.parseError;
        WScript.Echo("You have error " + myErr.reason);
    } else {
        xslproc = xslt.createProcessor();
        xslproc.input = xmldoc;
        xslproc.addParameter("param1", "Hello");
        xslproc.transform();
        WScript.Echo(xslproc.output);
    }
}
```

C/C++ Syntax

 Copy

```
HRESULT addParameter (BSTR baseName, VARIANT parameter, BSTR
namespaceURI);
```

Parameters

`baseName` [in]

The name that will be used inside the style sheet to identify the parameter context.

`parameter` [in]

A number, Boolean, string, node list, or node. Passing in a single node will produce a node list that contains one node (shortcut). For MSXML 6.0, to remove a parameter previously added to the processor, you can pass a value of VT_EMPTY, VT_NULL, or a NULL IDispatch or IUnknown instead. This acts as a signal to the processor to remove any previously added parameter of the same name. However for MSXML3.0, this function cannot remove previously added parameters.

`namespaceURI` [in, optional]

An optional namespace.

Return Values

E_FAIL if `readyState` is READYSTATE_INTERACTIVE.

Example

C++

 Copy

```
#include "stdio.h"
#include "msxml6.h"

int checkParseError(IXMLDOMParseErrorPtr pError);
void dump_com_error(_com_error &e);

int main(int argc, char* argv[])
{
    CoInitialize(NULL);
    HRESULT hr;

    try{
        BOOL bResult = FALSE;
        short sResult = FALSE;

        IXMLElement2Ptr pStyleSheet=NULL;
        IXSLTemplatePtr pIXSLTemplate=NULL;
        IXSLProcessorPtr pXSLProcessor=NULL;

        hr = pIXSLTemplate.CreateInstance(__uuidof(XSLTemplate60));

        hr=pStyleSheet.CreateInstance(__uuidof(FreeThreadedDOMDocument60));
        pStyleSheet->async = VARIANT_FALSE;
```

```

hr=pStyleSheet->load("sample.xls");
//check on the parser error
if(hr!=VARIANT_TRUE)
{
    return checkParseError(pStyleSheet->parseError);
}

pIXSLTemplate->stylesheet = pStyleSheet.GetInterfacePtr();
pXSLProcessor = pIXSLTemplate->createProcessor();

IXMLDOMDocumentPtr    pInputDoc;

hr = pInputDoc.CreateInstance(__uuidof(DOMDocument60));
pInputDoc->async = VARIANT_FALSE;
hr = pInputDoc->load("books.xml");
//check on the parser error
if(hr!=VARIANT_TRUE)
{
    return checkParseError(pInputDoc->parseError);
}

pInputDoc->async = VARIANT_FALSE;
pXSLProcessor->input = pInputDoc.GetInterfacePtr();

hr=pXSLProcessor->addParameter("param1", "Hello", "");

VARIANT_BOOL vtRet = pXSLProcessor->transform();
if(vtRet != VARIANT_TRUE)
{
    MessageBox(NULL, "transformation failed", "Error", MB_OK);
    return -1;
}
_bstr_t bstrOutput = pXSLProcessor->Getoutput();

MessageBox(NULL, bstrOutput, "Transformed Output", MB_OK);

}

catch(_com_error &e)
{
    dump_com_error(e);
}
return 0;
}

int checkParseError(IXMLDOMParseErrorPtr pError)
{
    _bstr_t parseError =_bstr_t("At line ")+ _bstr_t(pError->Getline()) +
    _bstr_t("\n") + _bstr_t(pError->Getreason());
    MessageBox(NULL,parseError, "Parse Error",MB_OK);
    return -1;
}

```

```
void dump_com_error(_com_error &e)
{
    printf("Error\n");
    printf("\a\tCode = %08lx\n", e.Error());
    printf("\a\tCode meaning = %s", e.ErrorMessage());
    _bstr_t bstrSource(e.Source());
    _bstr_t bstrDescription(e.Description());
    printf("\a\tSource = %s\n", (LPCSTR) bstrSource);
    printf("\a\tDescription = %s\n", (LPCSTR) bstrDescription);
}
```

Resource File

The examples in this topic use the following file.

Sample.xsl

XML

Copy

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <xsl:output method="html"/>
    <xsl:param name="param1"/>
    <xsl:template match="/">
        The parameter value was: <xsl:value-of select="$param1"/>
    </xsl:template>
</xsl:stylesheet>
```

Output

The examples in this topic output the following:

The parameter value was: Hello

Remarks

The `addParameter` method can be called on `transformNode` handlers and between `transform` calls (in asynchronous processing), and further processing will use the updated parameter. Added parameters are referenced by `<xsl:param>` within the style sheet.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXSLProcessor](#)

appendChild Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource File](#)

[Output](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Appends a new child node as the last child of the node.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNode.appendChild(newChild);
```

Parameters

`newChild`

An object. Address of the new child node to be appended at the end of the list of children belonging to this node.

Return Value

An object. Returns the new child node successfully appended to the list.

Example

The following script example creates a new `IXMLDOMNode` object, and then uses the `appendChild` method to append it to the document's list of children. This example uses the resource file listed later in this topic.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var newNode;
xmlDoc.async = false;
xmlDoc.load("appendChild.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    WScript.Echo("Before appendChild:\n" + root.xml + "\n");
    newNode = xmlDoc.createNode(1, "newChild", "");
    root.appendChild(newNode);
    WScript.Echo("After appendChild:\n" + root.xml);
}
```

Resource File

The JScript example uses the following file.

XSLT File: `appendChild.xml`

XML

 Copy

```
<?xml version='1.0'?>
<root>
    <firstChild/>
</root>
```

Output

The JScript example outputs the following in a message box.

Before `appendChild`:

`<root>`

`<firstChild/>`

```
</root>
```

After `appendChild`:

```
<root>
```

```
<firstChild/>
```

```
<newChild/></root>
```

C/C++ Syntax

 Copy

```
HRESULT appendChild(  
    IXMLDOMNode *newChild,  
    IXMLDOMNode **outNewChild);
```

Parameters

`newChild` [in]

The address of the new child node to be appended to the end of the list of children of this node.

`outNewChild` [out, retval]

The new child node successfully appended to the list. If Null, no object is created.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `newChild` parameter is Null.

`E_FAIL`

The value returned if an error occurs.

Remarks

Using this method is equivalent to calling `insertBefore(newChild, null)`. For more information, see [insertBefore Method](#).

If the `newChild` parameter has an existing parent, the node is automatically removed from that parent before being inserted into its new location.

A node retains its namespace information even when moved. Moving a node does not create a namespace declaration; declarations are added when retrieving the XML source (through the `save` method or the `xml` property) to ensure that all namespaces are properly declared.

When inserting a node tree under another node that has a different owner document, the `ownerDocument` property for each inserted node is changed to match the owner document of its new parent.

When moving a node tree to another document, the content of all entity reference nodes contained therein is updated to conform to the new document. If the new document does not declare an entity that was moved into it, the entity reference will have no children, and the old content is removed. Existing references to nodes under the entity reference are still valid, but the node whose parent previously was the entity reference now has a null parent.

MSXML 6.0 validates additions to the DOM only when the user explicitly calls [validate Method1](#). This means that you do not have to add nodes to the tree in the same order as they are defined in the schema. The implication is that there may be intermediate states between calls to `validate` where the DOM is invalid against the schema.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMAttribute](#) | [IXMLDOMCDATASection](#) | [IXMLDOMCharacterData](#) |
[IXMLDOMComment](#) | [IXMLDOMDocument-DOMDocument](#) |
[IXMLDOMDocumentFragment](#) | [IXMLDOMDocumentType](#) | [IXMLDOMEElement](#) |
[IXMLDOMEntity](#) | [IXMLDOMEntityReference](#) | [IXMLDOMNode](#) | [IXMLDOMNotation](#) |
[IXMLDOMProcessingInstruction](#) | [IXMLDOMText](#)

See Also

[save Method \(DOMDocument\)](#)

[xml Property1](#)

[ownerDocument Property](#)

appendData Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Appends the supplied string to the existing string data.

JScript Syntax

 Copy

```
oXMLDOMCharacterData.appendData(data);
```

Parameters

data

A string containing the data that is to be appended to the existing string.

Example

The following script example creates an `IXMLDOMComment` object and uses the `appendData` method to add text to the string.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var comment;
xmlDoc.async = false;
```

```
xmlDoc.loadXML("<root></root>");  
if (xmlDoc.parseError.errorCode != 0) {  
    var myErr = xmlDoc.parseError;  
    WScript.Echo("You have error " + myErr.reason);  
} else {  
    comment = xmlDoc.createComment("Hello...");  
    comment.appendData(" ... World!");  
    WScript.Echo(comment.data);  
}
```

Output

 Copy

```
Hello... . . . World!
```

C/C++ Syntax

 Copy

```
HRESULT appendData(  
    BSTR data);
```

Parameters

`data` [in]

The string data to be appended to the existing string.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value returned if the string is not appended.

E_FAIL

The value returned if an error occurs.

Remarks

The `length` property is also updated by this operation.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMCDATASection](#) | [IXMLDOMCharacterData](#) | [IXMLDOMComment](#) | [IXMLDOMText](#)

See Also

[length Property \(IXMLDOMCharacterData\)](#)

clone Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[Applies to](#)

Clones a copy of the current `IXMLDOMSelection`, with the same position and context.

JScript Syntax

 Copy

```
var NewObjXMLDOMSelection = objXMLDOMSelection.clone();
```

Example

jscript

 Copy

```
// Create a DOMDocument2 object and load some XML.
xmldoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
xmldoc.loadXML("<root><elem1>Hello</elem1><elem2>World!</elem2></root>");
if (xmldoc.parseError.errorCode != 0) {
    var myErr = xmldoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    // Create an XMLDOMSelection object from selected nodes.
    xpath = xmldoc.selectNodes("root/elem1");
    // Cache the XPath expression and context.
    xpath.expr = "root/elem1";
    xpath.context = xmldoc;
    // Clone the XMLDOMSelection object.
    xpath2 = xpath.clone();
    temp1 = xpath.peekNode(); // temp1 == <elem1/>
    WScript.Echo("temp1: " + temp1.xml);
    temp2 = xpath2.peekNode(); // temp2 == <elem2/>
```

```
WScript.Echo("temp2: " + temp2.xml);
// Note that the position and context are maintained.
}
```

Output

temp1: <elem1>Hello</elem1>
temp2: <elem1>Hello</elem1>

 Copy

C/C++ Syntax

```
HRESULT clone(IXMLDOMSelection ** ppNode);
```

 Copy

Parameters

`ppNode` [out, retval]

The returned copy of the `IXMLDOMSelection` object.

Return Values

`S_OK`

The value returned if the cloning is successful.

`E_INVALIDARG`

The value returned if the input argument is Null.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMSelection](#)

cloneNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

Applies to

See Also

Clones a new node.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNode.cloneNode(deep);
```

Parameters

`deep`

Boolean. A flag that indicates whether to recursively clone all nodes that are descendants of this node. If True, creates a clone of the complete tree below this node. If False, clones this node and its attributes only.

Return Value

Object. Returns the newly created clone node.

Example

The following script example clones a node, and then appends it as a child of the top-level node.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var currNode;
var MyNewNode;
xmlDoc.async = false;
xmlDoc.loadXML("<root><book/><AUTHOR/></root>")
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    currNode = root.childNodes.item(1);
    MyNewNode = currNode.cloneNode(true);
    root.appendChild(MyNewNode);
    WScript.Echo(xmlDoc.xml);
}
```

Output

```
<root><book/><AUTHOR/><AUTHOR/></root>
```

C/C++ Syntax

```
HRESULT cloneNode(
    VARIANT_BOOL deep,
    IXMLDOMNode **cloneRoot);
```

Parameters

deep [in]

A flag that indicates whether to recursively clone all nodes that are descendants of this node. If True, creates a clone of the complete tree below this node. If False, clones this node and its attributes only.

`cloneRoot [out, retval]`

A newly created clone node.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `cloneRoot` parameter is Null.

Remarks

The cloned node has the same property values as this node for the following properties:

`nodeName` property, `nodeValue` property, `nodeType` property, `parentNode` property,

`ownerDocument` property, and, if it is an element, `attributes` property. The value of the clone's `childNodes` property depends on the setting of the `deep` flag parameter.

ⓘ Note

If the node is the `DOMDocument` node, it is safer to clone the document using the `save` method, as follows.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMAttribute](#) | [IXMLDOMCDATASection](#) | [IXMLDOMCharacterData](#) |

[IXMLDOMComment](#) | [IXMLDOMDocument-DOMDocument](#) |

[IXMLDOMDocumentFragment](#) | [IXMLDOMDocumentType](#) | [IXMLDOMElement](#) |

[IXMLDOMEntity](#) | [IXMLDOMEntityReference](#) | [IXMLDOMNode](#) | [IXMLDOMNotation](#) |

[IXMLDOMProcessingInstruction](#) | [IXMLDOMText](#)

See Also

nodeName Property1

nodeValue Property

nodeType Property1

parentNode Property1

ownerDocument Property

attributes Property1

childNodes Property

createAttribute Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

Applies to

See Also

Creates a new attribute with the specified name.

JScript Syntax

 Copy

```
var objXMLDOMAttribute = oXMLDOMDocument.createAttribute(name);
```

Parameters

name

A string specifying the name of the new attribute object. This name is subsequently available as the new node's `nodeName` property.

Return Value

An object. Returns the new `IXMLDOMAttribute` object.

Example

The following script example creates a new attribute called `ID` and adds it to the attributes of the `DOMDocument` object.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var newAtt;
var namedNodeMap;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    newAtt = xmlDoc.createAttribute("ID");
    namedNodeMap = root.attributes;
    namedNodeMap.setNamedItem(newAtt);
    for (var i=0; i<namedNodeMap.length; i++) {
        WScript.Echo(namedNodeMap.item(i).xml);
    }
}
```

Output

```
id=""
```

C/C++ Syntax

```
HRESULT createAttribute(
    BSTR name,
    IXMLDOMAttribute **attribute);
```

Parameters

`name` [in]

The name of the new attribute object. This name is subsequently available as the new

node's `nodeName` property.

`attribute` [out, retval]

The address of the new `IXMLDOMAttribute` object.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `attribute` parameter is Null.

`E_FAIL`

The value returned if an error occurs.

Remarks

Creating an attribute with this method is the same as using `createNode` where the `type` parameter value is `NODE_ATTRIBUTE` and no namespace is specified.

You cannot create a namespace-qualified attribute using the `createAttribute` method.

Regardless of whether a namespace prefix is included in the `name` parameter*,* the `namespaceURI` property for the new attribute is set to an empty string, "". An attribute constructed as part of an XML document load operation will never have both a prefix and an empty namespace Uniform Resource Identifier (URI). You can only create a namespace-qualified attribute using the `createNode` method of the `DOMDocument`.

No data value is set for the attribute during the create operation. You can set the value by calling the `setAttribute` method of the element object.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property remains null. To associate the attribute with an element, call the `setAttributeNode` method of the `IXMLDOMElement` object.

Because the `parentNode` property of an attribute always returns a Null value, this property will not change after associating the new attribute with an element using the `setAttribute`

method.

The `nodeType` property has the value `NODE_ATTRIBUTE`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[createNode Method](#)

[namespaceURI Property \(IXMLDOMNode\)](#)

[setAttribute Method](#)

[ownerDocument Property](#)

[parentNode Property1](#)

[nodeType Property1](#)

[nodeName Property1](#)

[IXMLDOMElement](#)

[IXMLDOMAttribute](#)

createCDATASection Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

Applies to

See Also

Creates a CDATA section node that contains the supplied data.

JScript Syntax

 Copy

```
var objXMLDOMCDATASection = oXMLDOMDocument.createCDATASection(data);
```

Parameters

`data`

A string specifying the value to be supplied to the new `IXMLDOMCDATASection` object's `nodeValue` property.

Return Value

An object. Returns the new `IXMLDOMCDATASection` object.

Example

The following script example creates a new CDATA section node and appends it as the last child node of the `DOMDocument` object.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var CDATASEction;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    CDATASEction = xmlDoc.createCDATASection("Hello World!");
    root.appendChild(CDATASEction);
    WScript.Echo(root.xml);
}
```

Output

```
<root><child/></root>
```

C/C++ Syntax

```
HRESULT createCDATASection(
    BSTR data,
    IXMLDOMCDATASection **cdata);
```

Parameters

`data` [in]

The value to be supplied to the new CDATA section object's `nodeValue` property.

`cdata` [out, retval]

The address of the new `IXMLDOMCDATASection`.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `cdata` parameter is Null.

E_FAIL

The value returned if an error occurs.

Remarks

Creating a CDATA section with this method is the same as using `createNode` where the `type` parameter value is `NODE_CDATA_SECTION` and no namespace is specified. You cannot specify a namespace with the `createCDATASection` method.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property is set to Null. To add the new object, you must explicitly call one of the node insert methods, `insertBefore` method, `replaceChild` method, or `appendChild` method.

The new object's `nodeType` property has the value `NODE_CDATA_SECTION`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[createNode Method](#)

[ownerDocument Property](#)

[parentNode Property1](#)

[insertBefore Method](#)

[nodeType Property](#)

[nodeValue Property](#)

[replaceChild Method](#)

[appendChild Method](#)

[IXMLDOMCDATASection](#)

createComment Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

[Versioning](#)

Applies to

[See Also](#)

Creates a comment node that contains the supplied data.

JScript Syntax

 Copy

```
var objXMLDOMComment = oXMLDOMDocument.createComment(data);
```

Parameters

`data`

The string specifying the value to be supplied to the new `Comment` object's `nodeValue` property.

Return Value

An object. Returns the new `IXMLDOMComment`.

Example

The following script example creates a new comment node and appends it as the last child node of the `DOMDocument` object.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var comment;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    comment = xmlDoc.createComment("Hello World!");
    root.appendChild(comment);
    WScript.Echo(root.xml);
}
```

Output

```
<root><child/><!--Hello World! --></root>
```

C/C++ Syntax

```
HRESULT createComment(
    BSTR data,
    IXMLDOMComment **comment);
```

Parameters

data[in]

A value to be supplied to the new comment object's `nodeValue` property.

comment[out, retval]

The address of the new `IXMLDOMComment` object.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `comment` parameter is Null.

E_FAIL

The value returned if an error occurs.

Remarks

Creating a comment with this method is the same as using `createNode` where the `type` parameter value is `NODE_COMMENT` and no namespace is specified. You cannot specify a namespace with the `createComment` method.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property is set to Null. To add the new object, you must explicitly call one of the node insert methods, `insertBefore` method, `replaceChild` method, or `appendChild` method.

The new object's `nodeType` property has the value `NODE_COMMENT`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[createNode Method](#)

[ownerDocument Property](#)

[parentNode Property1](#)

[insertBefore Method](#)

[replaceChild Method](#)

[appendChild Method](#)

[nodeType Property](#)

[nodeValue Property](#)

[IXMLDOMComment](#)

createDocumentFragment Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Creates an empty `IXMLDOMDocumentFragment` object.

JScript Syntax

 Copy

```
var objXMLDOMDocumentFragment = oXMLDOMDocument.createDocumentFragment();
```

Return Value

An object. Returns the empty `IXMLDOMDocumentFragment` object.

Example

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var docFragment;
xmlDoc.async = false;
xmlDoc.loadXML("<root/>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
```

```
docFragment = xmlDoc.createDocumentFragment();
docFragment.appendChild(xmlDoc.createElement("node1"));
docFragment.appendChild(xmlDoc.createElement("node2"));
docFragment.appendChild(xmlDoc.createElement("node3"));
WScript.Echo("fragment: " + docFragment.xml);
xmlDoc.documentElement.appendChild(docFragment);
WScript.Echo("document: " + xmlDoc.xml);
}
```

Output

 Copy

```
fragment: <node1/><node2/><node3/>
document: <root></root>
```

C/C++ Syntax

 Copy

```
HRESULT createDocumentFragment(
    IXMLDOMDocumentFragment **docFrag);
```

Parameters

docFrag[out, retval]

The address of the empty `IXMLDOMDocumentFragment`.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `docFrag` parameter is Null.

E_FAIL

The value returned if an error occurs.

Remarks

Creating a document fragment with this method is the same as using `createNode` where the `type` parameter value is `NODE_DOCUMENT_FRAGMENT` and no namespace is specified. You cannot specify a namespace with the `createDocumentFragment` method.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property is set to Null. To add the new object, you must explicitly call one of the node insert methods, `insertBefore` method, `replaceChild` method, or `appendChild` method.

The `nodeType` property has the value `NODE_DOCUMENT_FRAGMENT`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[createNode Method](#)

[ownerDocument Property](#)

[parentNode Property1](#)

[nodeType Property1](#)

[insertBefore Method](#)

[replaceChild Method](#)

[appendChild Method](#)

[IXMLDOMDocumentFragment](#)

createElement Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

Applies to

See Also

Creates an element node using the specified name.

JScript Syntax

 Copy

```
var objXMLDOMElement = oXMLDOMDocument.createElement(tagName);
```

Parameters

`tagName`

A string specifying the name for the new element node. The name is case-sensitive. This name is subsequently available as the element node's `nodeName` property.

Return Value

An object. Returns the `IXMLDOMElement` object for the new element.

Example

The following script example creates an element called `NEW` and appends it to an `IXMLDOMNode` object. It then sets the text value of the element to 123.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var newElem;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    newElem = xmlDoc.createElement("NEW");
    root.childNodes.item(0).appendChild(newElem);
    root.childNodes.item(0).lastChild.text = "123";
    WScript.Echo(root.childNodes.item(0).xml);
}
```

Output

 Copy

```
<child><NEW>123</NEW></child>
```

C/C++ Syntax

 Copy

```
HRESULT createElement(
    BSTR tagName,
    IXMLDOMELEMENT **element);
```

Parameters

tagName[in]

The name for the new element node. It is case-sensitive. This name is subsequently available as the element node's `nodeName` property.

`element[out,retval]`

The address of the `IXMLDOMELEMENT` interface for the new element.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `element` parameter is Null.

`E_FAIL`

The value returned if an error occurs.

Remarks

Creating an element with this method is the same as using `createNode` where the `type` parameter value is `NODE_ELEMENT` and no namespace is specified.

You cannot create a namespace-qualified element using the `createElement` method.

Regardless of whether a namespace prefix is included in the `tagName` parameter*,* the `namespaceURI` property for the new element node is set to an empty string, `""`. An element node constructed as part of an XML document load operation will never have both a prefix and an empty namespace Uniform Resource Identifier (URI). You can only create a namespace-qualified element using the `createNode` method of the `DOMDocument` object.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property is set to Null. To add the new object, you must explicitly call one of the node insert methods, `insertBefore` method, `replaceChild` method, or `appendChild` method.

The `nodeType` property has the value `NODE_ELEMENT`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[createNode Method](#)

[namespaceURI Property \(IXMLDOMNode\)](#)

[ownerDocument Property](#)

[parentNode Property1](#)

[nodeName Property1](#)

[nodeType Property1](#)

[insertBefore Method](#)

[replaceChild Method](#)

[appendChild Method](#)

[IXMLDOMElement](#)

createEntityReference Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

Applies to

See Also

Creates a new `EntityReference` object.

JScript Syntax

 Copy

```
var objXMLDOMEntityReference = oXMLDOMDocument.createEntityReference(name);
```

Parameters

name

A string specifying the name of the entity referenced. This name is subsequently available as the new object's `nodeName` property.

Return Value

An object. Returns the new `IXMLDOMEntityReference` object.

Example

The following script example creates a new `IXMLDOMEntityReference` object and appends it to an `IXMLDOMNode` object.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var MyEntity;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    MyEntity = xmlDoc.createEntityReference("newRef");
    root.childNodes.item(0).appendChild(MyEntity);
    WScript.Echo(root.xml);
}
```

Output

```
<root><child>&newRef;</child></root>
```

C/C++ Syntax

```
HRESULT createEntityReference(
    BSTR name,
    IXMLDOMEntityReference **entityRef);
```

Parameters

name[in]

The name of the entity referenced. This name is subsequently available as the new object's `nodeName` property.

`entityRef[out, retval]`

The address of the new `IXMLDOMEntityReference` object.

C/C++ Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `entityRef` parameter is Null.

`E_FAIL`

The value returned if an error occurs.

Remarks

Creating an entity reference with this method is the same as using `createNode` where the `type` parameter value is `NODE_ENTITY_REFERENCE` and no namespace is specified. You cannot specify a namespace within the `name` parameter.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property is set to Null. To add the new object, you must explicitly call one of the node insert methods, `insertBefore` method, `replaceChild` method, or `appendChild` method.

The `nodeType` property has the value `NODE_ENTITY_REFERENCE`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

nodeName Property1

nodeType Property1

createNode Method

ownerDocument Property

parentNode Property1

insertBefore Method

replaceChild Method

appendChild Method

IXMLDOMEntityReference

createNode Method

10/27/2016 • 3 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Creates a node using the supplied type, name, and namespace.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMDocument.createNode(Type, name, namespaceURI);
```

Parameters

Type

A variant. A value that uniquely identifies the node type. This can be specified using either the integer value or the string value. For the complete list of values, see [XML DOM Enumerated Constants](#) or the description for the `name` parameter.

name

A string containing the value for the new node's `nodeName` property. The relationship between the `name` and `Type` parameters is summarized in the Remarks section of this topic.

namespaceURI

A string defining the namespace URI. If specified, the node is created in the context of the

`namespaceURI` parameter with the prefix specified on the node name. If the `name` parameter does not have a prefix, this is treated as the default namespace.

Return Value

Object. Returns the newly created node.

Example

The following script example creates an attribute node called `Sci-Fi` and adds it to the attributes for the `DOMDocument` object.

```
jscript Copy  
  
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");  
var MyNode;  
var namedNodeMap;  
xmlDoc.async = false;  
xmlDoc.loadXML("<root><child/></root>");  
MyNode = xmlDoc.createNode(2, "Sci-Fi", "");  
namedNodeMap = xmlDoc.documentElement.childNodes.item(0).attributes;  
namedNodeMap.setNamedItem(MyNode);  
WScript.Echo(xmlDoc.documentElement.xml);
```

Output

```
Copy  
  
<root><child Sci-Fi="" /></root>
```

C/C++ Syntax

```
Copy  
  
HRESULT createNode(  
    VARIANT Type,  
    BSTR name,  
    BSTR namespaceURI,  
    IXMLDOMNode **node);
```

Parameters

Type [in]

The value that uniquely identifies the node type. This can be specified using either the integer value or the string value. For the complete list of values, see the [XML DOM Enumerated Constants](#) or the description given for the `name` parameter.

name [in]

The value for the new node's `nodeName` property. The relationship between the `name` and `Type` parameters is summarized in the **Remarks** section of this topic.

namespaceURI [in]

The namespace URI. If specified, the node is created in the context of the `namespaceURI` parameter with the prefix specified on the node name. If the `name` parameter does not have a prefix, this is treated as the default namespace.

node [out, retval]

The newly created node.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `node` parameter is Null.

E_FAIL

The value returned if an error occurs.

Remarks

The `name` parameter depends on the value of the `Type` parameter:

NODE_ATTRIBUTE

The name of the attribute.

NODE_CDATA_SECTION,

The `nodeName` property for these node types is a constant value; the `name` parameter is ignored.

NODE_COMMENT, NODE_DOCUMENT,

NODE_DOCUMENT_FRAGMENT,

NODE_TEXT

<code>NODE_DOCUMENT_TYPE</code>	The name of the document type; for example, the xxx in <code><!DOCTYPE xxx ...></code> .
<code>NODE_ELEMENT</code>	The name of the XML tag, with any namespace prefix included if present.
<code>NODE_ENTITY</code>	The name of the entity.
<code>NODE_ENTITY_REFERENCE</code>	The name of the entity referenced. The name does not include the leading ampersand or the trailing semicolon. The name includes the namespace, if one is present.
<code>NODE_NOTATION</code>	The name of the notation.
<code>NODE_PROCESSING_INSTRUCTION</code>	The target, the first token following the <code><?</code> characters.

You cannot create a node of type `NODE_DOCUMENT`, `NODE_DOCUMENT_TYPE`, `NODE_ENTITY`, or `NODE_NOTATION`.

When a node is created, it is created in the context of a namespace if one is supplied (if the `namespaceURI` parameter is supplied). If one is not supplied, the node is created in the namespace of the document. If the `namespaceURI` parameter is specified, the node is created in the context of the `namespaceURI` parameter with the prefix specified on the node name.

For node types that do not have names, the empty string, "", should be passed as the `name` parameter.

For elements and entity references, when the value of the `namespaceURI` parameter is anything other than "", and the value of the `name` parameter does not contain a prefix (xxx in xxx:yyy), the value of the `namespaceURI` parameter is treated as the default namespace.

Attributes cannot be scoped to a default namespace, and other elements are not qualified to a particular namespace; they are treated as being from the namespace defined by the document itself.

When the value of `namespaceURI` parameter is the empty string, "", the node is created without a namespace. Creating a qualified node without specifying a nonempty `namespaceURI` returns an error. This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[nodeName Property1](#)

createProcessingInstruction Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Creates a processing instruction node that contains the supplied target and data.

JScript Syntax

 Copy

```
var objXMLDOMProcessingInstruction =  
oXMLDOMDocument.createProcessingInstruction(target, data);
```

Parameters

`target`

A string specifying the target part of the processing instruction. This supplies the `nodeName` property of the new object.

`data`

A string specifying the rest of the processing instruction preceding the closing ?> characters. This supplies the `nodeValue` property for the new object.

Return Value

An object. Returns the new `IXMLDOMProcessingInstruction` object.

Example

The following script example specifies the target string "xml" and the data string "version=\"1.0\""`"` to generate the processing instruction `<?XML version="1.0"?>`.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var pi;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    pi = xmlDoc.createProcessingInstruction("xml", "version=\"1.0\"");
    xmlDoc.insertBefore(pi, xmlDoc.childNodes.item(0));
    WScript.Echo(xmlDoc.xml);
}
```

Output

 Copy

```
<?xml version="1.0" ?>
<root><child/></root>
```

C/C++ Syntax

 Copy

```
HRESULT createProcessingInstruction(
    BSTR target,
    BSTR data,
    IXMLDOMProcessingInstruction **pi);
```

Parameters

`target` [in]

The target part of the processing instruction. It supplies the `nodeName` property of the new

object.

`data` [in]

The remainder of the processing instruction preceding the closing ?> characters. It supplies the `nodeValue` property for the new object.

`pi` [out, retval]

The address of the new `IXMLDOMProcessingInstruction` object.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `pi` parameter is Null.

`E_FAIL`

The value returned if an error occurs.

Remarks

Creating a processing instruction node with this method is the same as using `createNode` where the `Type` parameter value is `NODE_PROCESSING_INSTRUCTION` and no namespace is specified. You cannot specify a namespace with the `createProcessingInstruction` method.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property is set to Null. To add the new object, you must explicitly call one of the node insert methods, `insertBefore` method, `replaceChild` method, or `appendChild` method.

The new object's `nodeType` property has the value `NODE_PROCESSING_INSTRUCTION`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[nodeName Property1](#)

[nodeType Property1](#)

[nodeValue Property](#)

[IXMLDOMProcessingInstruction](#)

[createNode Method](#)

[ownerDocument Property](#)

[insertBefore Method](#)

[replaceChild Method](#)

[appendChild Method](#)

createProcessor Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource File](#)

[Output](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

Creates a rental-model `IXSLProcessor` object that will use this template.

JScript Syntax

 Copy

```
var objXSLProcessor = objXSLTemplate.createProcessor();
```

Example

JavaScript

 Copy

```
var xslt = new ActiveXObject("Msxml2.XSLTemplate.3.0");
var xslDoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.3.0");
var xslProc;
xslDoc.async = false;
xslDoc.load("createProcessor.xsl");
if (xslDoc.parseError.errorCode != 0) {
    var myErr = xslDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xslt.stylesheet = xslDoc;
    var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
    xmlDoc.async = false;
    xmlDoc.load("books.xml");
```

```
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xslProc = xslt.createProcessor();
    xslProc.input = xmlDoc;
    xslProc.addParameter("param1", "Hello");
    xslProc.transform();
    WScript.Echo(xslProc.output);
}
}
```

Resource File

The JScript and Visual Basic examples use the following file.

createProcessor.xsl

XML

Copy

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <xsl:output method="html"/>
    <xsl:param name="param1"/>
    <xsl:template match="/">
        The parameter value was: <xsl:value-of select="$param1"/>
    </xsl:template>
</xsl:stylesheet>
```

Output

The JScript and Visual Basic examples output the following.

The parameter value was: Hello

C/C++ Syntax

```
HRESULT createProcessor(IXSLProcessor** ppProcessor);
```

Parameters

ppProcessor[out, retval]

The returned processor associated with this template.

Return Values

E_OUTOFMEMORY

E_FAIL

The value returned if the template has no style sheet.

Remarks

Multiple processors can be created from the same `IXSLTemplate` object.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXSLTemplate](#)

createTextNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Visual Basic Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Creates a text node that contains the supplied data.

JScript Syntax

 Copy

```
var objXMLDOMText = oXMLDOMDocument.createTextNode(data);
```

Parameters

data

A string specifying the value to be supplied to the new text object's `nodeValue` property.

Return Value

An object. Returns the new `IXMLDOMText` object.

Example

The following script example creates an `IXMLDOMText` object, and then uses the `insertBefore` method to place it before the first child node of the document.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
var root;
var MyText;
var MyNewNode;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    MyText = xmlDoc.createTextNode("Hello World");
    MyNewNode = root.insertBefore(MyText, root.childNodes.item(0));
    WScript.Echo(xmlDoc.xml);
}
```

Output

```
<root>Hello World!<child/></root>
```

Visual Basic Syntax

```
Set objXMLDOMText = oXMLDOMDocument.createTextNode
(data)
```

Parameters

`data`

A string specifying the value to be supplied to the new text object's `nodeValue` property.

Return Value

An object. Returns the new `IXMLDOMText` object.

Example

The following Microsoft® Visual Basic® example creates an `IXMLDOMText` object, and then uses the `insertBefore` method to place it before the first child node of the document.

```
VB Copy  
  
Dim xmlDoc As New Msxml2.DOMDocument30  
Dim root As IXMLDOMELEMENT  
Dim MyText As IXMLDOMText  
Dim MyNewNode As IXMLDOMNode  
xmlDoc.async = False  
xmlDoc.loadXML "<root><child/></root>"  
If (xmlDoc.parseError.errorCode <> 0) Then  
    Dim myErr  
    Set myErr = xmlDoc.parseError  
    MsgBox("You have error " & myErr.reason)  
Else  
    Set root = xmlDoc.documentElement  
    Set MyText = xmlDoc.createTextNode("Hello World")  
    Set MyNewNode = root.insertBefore(MyText, root.childNodes.Item(0))  
    MsgBox (xmlDoc.xml)  
End If
```

Output

```
<root>Hello World!<child/></root>
```

C/C++ Syntax

```
HRESULT createTextNode(  
    BSTR data,  
    IXMLDOMText **text);
```

Parameters

`data` [in]

The value to be supplied to the new text's `nodeValue`.

`text` [out, retval]

The address of new `IXMLDOMText`.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `text` parameter is Null.

`E_FAIL`

The value returned if an error occurs.

Remarks

Creating a text node with this method is the same as using `createNode` where the `Type` parameter value is `NODE_TEXT` and no namespace is specified. You cannot specify a namespace with the `createTextNode` method.

Although this method creates the new object in the context of this document, it does not automatically add the new object to the document tree. (In other words, although the `ownerDocument` property of the new node points to this document object, the `parentNode` property is set to Null.) To add the new object, you must explicitly call one of the node insert methods, `insertBefore` method, `replaceChild` method, or `appendChild` method.

The new `nodeType` property has the value `NODE_TEXT`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[nodeValue Property](#)

[IXMLDOMText](#)

[createNode Method](#)

[ownerDocument Property](#)

[parentNode Property1](#)

[nodeType Property1](#)

[insertBefore Method](#)

[replaceChild Method](#)

[appendChild Method](#)

errorParameters Method

10/27/2016 • 2 minutes to read

In this article

- [JScript Syntax](#)
- [C/C++ Syntax Using Smart Pointers](#)
- [Parameters](#)
- [Return Values](#)
- [Example](#)
- [Applies to](#)
- [Versioning](#)

[This sample code uses features that were implemented only in MSXML 6.0.]

Returns the parameter value for a given index. This parameter value is a string that is used in the construction of the error message. For example, when a validation error is reported on a missing element, the first error parameter, `errorParameters(0)`, shows where the missing element is located and the second error parameter, `errorParameters(1)`, shows which element is missing.

Note

The error parameters returned from this method are specific to each error message. An error message might not have any error parameters.

JScript Syntax

 Copy

```
strParam = objXMLDOMParseError2.errorParameters(index);
```

C/C++ Syntax Using Smart Pointers

 Copy

```
_bstr_t strParam = objXMLDOMParseError2->errorParameters (index);
HRESULT errorParameters (
    LONG index,
    BSTR *strParam);
```

Parameters

`index` [in]

A long integer representing the index of an error parameter.

`strParam` [out,retval]

A BSTR string containing the error parameter of the given index.

Return Values

`S_OK`

The error parameter has been retrieved successfully.

`E_FAIL`

The index is out of bounds and the resultant `strParam` parameter is NULL.

Example

This example uses the same two resource files used in the allErrors example, books.xml and books.xsd. We've provided source files for the sample in three languages: JScript, Visual Basic, and C++. The output is the same in each language.

- [Resource Files \(books.xml and books.xsd\)](#)
- [JScript Code \(errorParams.js\)](#)
- [C/C++ Code \(errorParams.cpp\)](#)
- [Output for the errorParams Example](#)

Applies to

[IXMLDOMParseError2 Interface](#)

Versioning

Implemented in: MSXML 6.0

deleteData Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Deletes specified data.

JScript Syntax

 Copy

```
oXMLDOMCharacterData.deleteData(offset, count);
```

Parameters

`offset`

A long integer value specifying the offset, in characters, at which to start deleting string data.

`count`

A long integer value specifying the number of characters to delete.

Example

The following script example creates an `IXMLDOMComment` object and uses the `deleteData` method to delete three characters of data starting after the third character in data for the `IXMLDOMComment` object.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
var comment;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    comment = xmlDoc.createComment("123456789");
    WScript.Echo("Before deleting data: " + comment.xml);
    comment.deleteData(3, 3);
    WScript.Echo("After deleting data: " + comment.xml);}
```

Output

Before deleting data: <!--123456789>

After deleting data: <!--123789>

C/C++ Syntax

```
HRESULT deleteData(
    long offset,
    long count);
```

Parameters

offset[in]

The offset, in characters, at which to start deleting string data.

count[in]

The number of characters to delete.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value when returning Null.

E_FAIL

The value returned if an error occurs.

Remarks

If the offset and count of characters specify a range beyond the end of the string, this method deletes only to the end of the string.

The `length` property is updated by this operation.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMCDATASection](#) | [IXMLDOMCharacterData](#) | [IXMLDOMComment](#) | [IXMLDOMText](#)

See Also

[length Property \(IXMLDOMCharacterData\)](#)

Dll SetProperty Method

10/27/2016 • 2 minutes to read

In this article

[Syntax](#)

[Arguments](#)

[Return Value](#)

[Remarks](#)

[Versioning](#)

This method is a globally exported function in the MSXML DLL. This method sets a global property for the DLL.

Syntax

 Copy

```
HRESULT Dll SetProperty(DWORD dwPropertyID, LPVOID pv);
```

Arguments

`dwPropertyID`

The property ID to set.

`pv`

An [in] argument that specifies the property value.

Return Value

Returns S_OK if no error is generated.

Remarks

This function is used to set global properties for the DLL.

The only value supported for `dwPropertyID` is 1. Using this property ID, you can set a callback that will be called when the physical memory for nodes is released. On Windows 7, Windows Server 2008 R2, Windows Vista, Windows Server 2008, and Windows XP, the callback is called when the internal garbage collector of MSXML finalizes the nodes. On Windows 8, the callback is called immediately after nodes are released.. The signature of the callback is:

```
void NodeFinalizeCallback(IUnknown* pUnknown);
```

The following example shows the use of this method to register a callback that is called when nodes are finalized:

```
c++ Copy
#include <windows.h>
#include <dispex.h>
#include <urlmon.h>
#include <stdio.h>
#include <msxml6.h>

#define CHK(stmt) do{ hr=(stmt); if (FAILED(hr)) {printf("FAILED at " #stmt); goto CleanUp;} }while(0)
#define SAFE_RELEASE(p) do{ if ((p)) { (p)->Release(); (p) = NULL; } }while(0)

void NodeFinalizeCallback(IUnknown* pUnknown)
{
    printf("Finalize: %p\r\n", pUnknown);
}

typedef HRESULT (*LPFNDDll SetProperty)(DWORD dw, LPVOID pv);

int __cdecl main(int argc, char ** argv)
{
    HRESULT hr = S_OK;
    CHK( CoInitialize(NULL));
    {
        VARIANT_BOOL isSuccessfull;
        BSTR xml = NULL;
        IXMLDOMDocument3* pDoc = NULL;
        IXMLDOMNode* pN = NULL;
        IXMLDOMNode* pN2 = NULL;
        IXMLDOMELEMENT* pElem = NULL;

        CHK( ::CoCreateInstance(CLSID_DOMDocument60, NULL, CLSCTX_ALL,
IID_IXMLDOMDocument3, (void**)&pDoc) );
        HMODULE hMsxml6 = ::GetModuleHandleW(L"msxml6.dll");
        if (NULL==hMsxml6)
        {
```

```

        printf("unexpected: Creating DOMDocument60 object should have loaded
msxml6.dll");
        goto CleanUp;
    }
    LPFNDll SetProperty Dll SetProperty = (LPFNDll SetProperty)
GetProcAddress(hMsxml6, "Dll SetProperty");
    if (NULL==Dll SetProperty)
    {
        printf("This version of Msxml6 does not support Dll SetProperty. You
need at least Msxml6 SP2.");
        goto CleanUp;
    }
    Dll SetProperty(1, NodeFinalizeCallback);

    CHK(pDoc->get_documentElement(&pElem));

    VARIANT v;
    V_VT(&v)=VT_I4;
    V_I4(&v)=NODE_DOCUMENT_FRAGMENT;

    CHK(pDoc->createNode(v, L"xmlns:x", NULL, &pN));
    {
        IUnknown* pU = NULL;
        pN->QueryInterface(IID_IUnknown, (void**)&pU);
        printf("Create: %p\r\n", pU);
        SAFE_RELEASE(pU);
    }
    SAFE_RELEASE(pDoc);

    CHK( pN->selectSingleNode(L".", &pN2));

CleanUp:
    SAFE_RELEASE(pElem);
    SAFE_RELEASE(pDoc);
    SAFE_RELEASE(pN2);
    SAFE_RELEASE(pN);
}

CoUninitialize();
if(FAILED(hr))
    printf("failed");
else
    printf("done");
return 0;
}

```

Versioning

Implemented in: MSXML 6.0 SP2 and later.

get Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource File](#)

[Output](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

This method is deprecated in MSXML 6.0, where it throws a Not Implemented exception.

Instead of using this method in MSXML 6.0, use the [getSchema Method1](#).

Returns a read-only XML Document Object Model (DOM) node that contains the `<Schema>` element.

JScript Syntax

 Copy

```
var objXMLDOMNode = objXMLDOMSchemaCol.get(namespaceURI);
```

Parameters

`namespaceURI`

The namespace URI associated with the schema to return.

This can be any string that can be used in an `xmlns` attribute, but it cannot contain entity references. The same white space normalization that occurs on the `xmlns` attribute also occurs on this argument (that is, leading and trailing white space is trimmed, new lines are

converted to spaces, and multiple adjacent white space characters are collapsed into one space).

Example

jscript

 Copy

```
var oSchemaCache = new ActiveXObject("Msxml2.XMLSchemaCache.3.0");
var nsTarget = "";
oSchemaCache.add(nsTarget, "rootChild.xdr");
var oDOMNode = oSchemaCache.get(nsTarget);
WScript.Echo(oDOMNode.xml);
```

Resource File

The JScript example in this topic uses the following file:

rootChild.xdr

XML

 Copy

```
<?xml version="1.0"?>
<Schema xmlns="urn:schemas-microsoft-com:xml-data">
  <ElementType name="child"/>
  <ElementType name="root" model="closed">
    <element type="child"/>
  </ElementType>
</Schema>
```

Output

The example in this topic outputs the contents of the rootChild.xdr schema.

C/C++ Syntax

```
HRESULT get(BSTR namespaceURI, IXMLDOMNode ** schemaNode);
```

Parameters

namespaceURI [in]

The namespace URI associated with the schema to return.

This may be any string that can be used in an `xmlns` attribute, but it cannot contain entity references. The same white space normalization that occurs on the `xmlns` attribute also occurs on this argument (that is, leading and trailing white space is trimmed, new lines are converted to spaces, and multiple adjacent white space characters are collapsed into one space).

schemaNode [out, retval]

A read-only `IXMLDOMNode` that represents the schema that is returned.

Return Values

S_OK

The value returned if successful.

Remarks

This will not necessarily be the same document object provided by the `add` method, because the `add` method might have copied the schema. For inline schemas, this will apply directly to the `<Schema>` node embedded within the document.

Versioning

Implemented in: MSXML 3.0. Not implemented in MSXML 6.0.

Applies to

[IXMLDOMSchemaCollection-XMLSchemaCache](#)

See Also

[add Method \(IXMLDOMSchemaCollection-XMLSchemaCache\)](#)

getAllResponseHeaders Method (IXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the values of all the HTTP headers.

JScript Syntax

 Copy

```
strValue = oXMLHttpRequest.getAllResponseHeaders();
```

Return Value

A string. Contains the resulting header information.

Example

jscript

 Copy

```
var xmlhttp = new ActiveXObject("Msxml2.XMLHTTP.6.0");
xmlhttp.open("GET", "http://localhost/sample.xml", false);
xmlhttp.send();
WScript.Echo(xmlhttp.getAllResponseHeaders());
```

Output

This example returns the resulting page header information that was returned by the Web (HTTP) server hosting the specified page (sample.xml). For example, your output should include the following type header field information:

 Copy

```
Server:Microsoft-IIS/5.1
X-Powered-By:ASP.NET
Date:Sat, 07 Jun 2003 23:23:06 GMT
Content-Type:text/xml
Accept-Ranges:bytes
Last Modified:Sat, 06 Jun 2003 17:19:04 GMT
ETag:"a0e2eeba4f2cc31:97f"
Content-Length:9
```

C/C++ Syntax

 Copy

```
HRESULT getAllResponseHeaders(BSTR *pbstrHeaders);
```

Parameters

`pbstrHeaders` [out, retval]

The resulting header information.

Return Values

S_OK

The value returned if successful.

Example

C++

 Copy

```
HRESULT hr;
BSTR bstrValue = NULL;
IXMLHttpRequest *pIXMLHttpRequest = NULL;
```

```
try
{
    // Create XMLHttpRequest object and initialize pIXMLHttpRequest.
    hr = pIXMLHttpRequest->getAllResponseHeaders(&bstrValue);
    if(SUCCEEDED(hr))
        ::MessageBox(NULL, bstrValue, _T("All Response Headers"), MB_OK);
}
catch(...)
{
    DisplayErrorToUser();
}
// Release pIXMLHttpRequest when finished with it.
```

Remarks

Each header name/value pair is separated by a combination carriage return-line feed character (vbCrLf in Microsoft® Visual Basic®).

The results of this method are valid only after the `send` method has been successfully completed.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLHTTPRequest](#)

See Also

[send Method \(IXMLHTTPRequest\)](#)

[getResponseHeader Method \(IXMLHTTPRequest\)](#)

[setRequestHeader Method \(IXMLHTTPRequest\)](#)

getAllResponseHeaders Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the values of all the HTTP headers.

JScript Syntax

 Copy

```
strValue = oServerXMLHTTPRequest.getAllResponseHeaders();
```

Example

jscript

 Copy

```
var xmlServerHttp = new ActiveXObject("Msxml2.ServerXMLHTTP.6.0");
xmlServerHttp.open("GET", "http://localhost/sample.xml", false);
xmlServerHttp.send();
WScript.Echo(xmlServerHttp.getAllResponseHeaders());
```

Output

This example returns the resulting page header information that was returned by the Web (HTTP) server hosting the specified page (sample.xml). For example, your output should include the following type header field information:

 Copy

```
Server: Microsoft-IIS/5.1
Connection: keep-alive
Date: Mon, 09 Jun 2003 18:20:29 GMT
Content-Type: text/xml
Accept-Ranges: bytes
Last Modified: Fri, 06 Jun 2003 12:11:56 GMT
ETag: "1033b5d2242cc31:97c"
Content-Length: 4550
```

C/C++ Syntax

 Copy

```
HRESULT getAllResponseHeaders(BSTR *pbstrHeaders);
```

Parameters

`pbstrHeaders` [out, retval]

The resulting header information.

Return Values

`S_OK`

The value returned if successful.

Remarks

Each header name/value pair is separated by a combination carriage return–line feed character (vbCrLf in Microsoft® Visual Basic®).

The results of this method are valid only after the `send` method has been successfully completed.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IServerXMLHTTPRequest](#)-[ServerXMLHTTP](#)

See Also

[send Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[getResponseHeader Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[setRequestHeader Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

getAllResponseHeaders Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the values of all the HTTP headers.

JScript Syntax

 Copy

```
strValue = oServerXMLHTTPRequest.getAllResponseHeaders();
```

Example

jscript

 Copy

```
var xmlServerHttp = new ActiveXObject("Msxml2.ServerXMLHTTP.6.0");
xmlServerHttp.open("GET", "http://localhost/sample.xml", false);
xmlServerHttp.send();
WScript.Echo(xmlServerHttp.getAllResponseHeaders());
```

Output

This example returns the resulting page header information that was returned by the Web (HTTP) server hosting the specified page (sample.xml). For example, your output should include the following type header field information:

 Copy

```
Server: Microsoft-IIS/5.1
Connection: keep-alive
Date: Mon, 09 Jun 2003 18:20:29 GMT
Content-Type: text/xml
Accept-Ranges: bytes
Last Modified: Fri, 06 Jun 2003 12:11:56 GMT
ETag: "1033b5d2242cc31:97c"
Content-Length: 4550
```

C/C++ Syntax

 Copy

```
HRESULT getAllResponseHeaders(BSTR *pbstrHeaders);
```

Parameters

`pbstrHeaders` [out, retval]

The resulting header information.

Return Values

`S_OK`

The value returned if successful.

Remarks

Each header name/value pair is separated by a combination carriage return–line feed character (vbCrLf in Microsoft® Visual Basic®).

The results of this method are valid only after the `send` method has been successfully completed.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IServerXMLHTTPRequest](#)-[ServerXMLHTTP](#)

See Also

[send Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[getResponseHeader Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[setRequestHeader Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

getAttribute Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Gets the value of the attribute.

JScript Syntax

 Copy

```
objValue = oXMLDOMELEMENT.getAttribute(name);
```

Parameters

`name`

A string specifying the name of the attribute to return.

Return Value

A variant. Returns the value as a string if the attribute value is a non-empty string. Returns Null if the named attribute does not have a specified value, or an implied default value, such as one taken from a DTD or schema.

Example

`jscript`

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, sIdValue;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    sIdValue = nodeBook.getAttribute("id")
    WScript.Echo(sIdValue);
}
```

Output

When used with the [sample XML file \(books.xml\)](#), this example returns "bk101", the value of the `id` attribute for the first instance of the `<book>` element in books.xml.

C/C++ Syntax

 Copy

```
HRESULT getAttribute(
    BSTR name,
    VARIANT *value);
```

Parameters

`name` [in]

The name of the attribute to return.

`value` [out, retval]

The string that contains the attribute value. The empty string is returned if the named attribute does not have a default value specified.

Return Values

`S_OK`

The value returned if successful.

S_FALSE

The value when no attribute with the given name is found.

E_INVALIDARG

The value returned if the `name` parameter is Null.

Example

```
c++ Copy
BOOL DOMElementAttribute()
{
    BOOL bResult = FALSE;
    _variant_t varValue;
    BSTR bstrAttributeName = ::SysAllocString(_T("dateCreated"));
    IXMLDOMDocument *pIXMLDOMDocument = NULL;
    IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;
    HRESULT hr;

    try
    {
        // Create an instance of DOMDocument and initialize
        // pIXMLDOMDocument.
        // Load/create an XML fragment.
        hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
        SUCCEEDED(hr) ? 0 : throw hr;

        If(pIXMLDOMELEMENT)
        {
            varValue = _T("year 2000");
            hr = pIXMLDOMELEMENT->setAttribute(bstrAttributeName, varValue);
            SUCCEEDED(hr) ? 0 : throw hr;

            hr = pIXMLDOMELEMENT->getAttribute(bstrAttributeName, &varValue);
            SUCCEEDED(hr) ? 0 : throw hr;
            if(varValue.vt != VT_NULL)
            {
                ::MessageBox(NULL, _bstr_t(varValue), bstrAttributeName, MB_OK);
                bResult = TRUE;
            }
            ::SysFreeString(bstrAttributeName);
            bstrAttributeName = NULL;
            pIXMLDOMELEMENT->Release();
        }
    }
    catch(...)
    {
        if(bstrAttributeName)
            ::SysFreeString(bstrAttributeName);
        if(pIXMLDOMELEMENT)
```

```
    pIXMLDOMELEMENT->Release();
    DisplayErrorToUser();
}
return bResult;
}
```

Remarks

You can also retrieve attributes by using the `getNamedItem` method of the `IXMLDOMNodeMap`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMELEMENT](#)

See Also

[getNamedItem Method](#)

[IXMLDOMNodeMap](#)

getAttributeNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Gets the attribute node.

JScript Syntax

 Copy

```
var objXMLDOMAttribute = oXMLDOMElement.getAttributeNode(name);
```

Parameters

`name`

The string specifying the name of the attribute to be retrieved.

Return Value

An object. Returns `IXMLDOMAttribute` with the supplied name, or Null if the named attribute cannot be found on this element.

Example

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, nodeId, sIdValue;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodeId = nodeBook.getAttributeNode("id");
    sIdXml = nodeId.xml;
    WScript.Echo(sIdXml);
}
```

Output

When used with the [sample XML file \(books.xml\)](#), this example returns the full attribute xml string of:

 Copy

```
id="bk101"
```

for the `id` attribute for the first instance of the `<book>` element in books.xml.

C/C++ Syntax

 Copy

```
HRESULT getAttributeNode(
    BSTR name,
    IXMLDOMAttribute **attributeNode);
```

Parameters

`name` [in]

The name of the attribute to be retrieved.

`attributeNode` [out, retval]

An `IXMLDOMAttribute` object that is returned with the supplied name, or Null if the named attribute cannot be found on this element.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value returned when no attribute with the given name is found.

E_INVALIDARG

The value returned if `name` is Null.

Example

```
c++ Copy
BOOL DOMElementAttributeNode()
{
    BOOL bResult = FALSE;
    BSTR bstrAttributeName = ::SysAllocString(_T("dateCreated"));
    IXMLDOMAttribute* pIXMLDOMAttribute = NULL;
    IXMLElement *pIXMLDOMELEMENT = NULL;
    IXMLDOMDocument *pIXMLDOMDocument = NULL;
    HRESULT hr;

    try
    {
        // Create an instance of DOMDocument and initialize
        // pIXMLDOMDocument.
        // Load/create an XML fragment.
        hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
        SUCCEEDED(m_hr) ? 0 : throw hr;

        If(pIXMLDOMELEMENT)
        {
            hr = pIXMLDOMELEMENT->getAttributeNode(bstrAttributeName,
&pIXMLDOMAttribute);
            if(SUCCEEDED(hr) && pIXMLDOMAttribute)
            {
                // Read node value and display . . .
                bResult = TRUE;
                pIXMLDOMAttribute->Release();
                pIXMLDOMAttribute = NULL;
            }
            ::SysFreeString(bstrAttributeName);
            bstrAttributeName = NULL;
            pIXMLDOMELEMENT->Release();
            pIXMLDOMELEMENT = NULL;
        }
    }
}
```

```
catch(...)  
{  
    if(bstrAttributeName)  
        ::SysFreeString(bstrAttributeName);  
    if(pIXMLDOMAttribute)  
        pIXMLDOMAttribute->Release();  
    if(pIXMLDOMELEMENT)  
        pIXMLDOMELEMENT->Release();  
    DisplayErrorToUser();  
}  
return bResult;  
}
```

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMELEMENT](#)

See Also

[IXMLDOMAttribute](#)

getDeclaration Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource Files](#)

[Output](#)

[C/C++ Syntax](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Returns an `ISchemaItem` object. This object is the declaration of the DOM node that is sent to the function. The declaration specifies the schema information that is used to validate the document item in the DOM node. The `getDeclaration` method also includes the declarations of XML schemas used by the associated document. This provides access to XML schemas loaded using `xsi:schemaLocation`, as well as XML schemas loaded from a schema cache object. The `ISchemaItem` interface can be used to obtain further information about the schema declaration stored in the `ISchemaItem` object.

JScript Syntax

 Copy

```
var oSchemaItem = oDom.namespaces.getDeclaration(oDOMNode);
```

Parameters

`oDOMNode`

An object. The DOM node for which a declaration object will be returned.

Return Values

`oSchemaItem`

An object. The schema item object that describes the specified DOM node, passed in the `node` parameter.

Example

The following JScript example shows the `getDeclaration` method in a schema.

```
jscript Copy  
  
// Load the XML document.  
var xmldoc = new ActiveXObject("MSXML2.DOMDocument.6.0");  
xmldoc.async = false;  
xmldoc.validateOnParse = false;  
xmldoc.load("doc.xml");  
xmldoc.setProperty("SelectionLanguage", "XPath");  
  
if (xmldoc.parseError.errorCode != 0){  
    var myErr = xmldoc.parseError;  
    WScript.Echo("You have error " & myErr.reason);  
} else {  
    // Retrieve the namespace URI for schema  
    // used in XML document.  
    var oDecl = xmldoc.namespaces.getDeclaration(xmldoc.documentElement);  
    WScript.Echo(oDecl.namespaceURI);  
}
```

Resource Files

The JScript and example uses the following files.

doc.xml

```
XML Copy  
  
<?xml version="1.0"?>  
<x:doc xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'  
       xmlns:x='http://xsdtesting'  
       xsi:schemaLocation='http://xsdtesting doc.xsd'>  
    <x:notDeclared/>  
</x:doc>
```

doc.xsd

XML

 Copy

```
<?xml version="1.0"?>
<xsschema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace = "http://xsdtesting"
    xmlns:x = "http://xsdtesting"
    elementFormDefault = "qualified">
    <xss:element name="doc"></xss:element>
</xsschema>
```

Output

When used with the sample XML (doc.xml) and schema file (doc.xsd) files above, the examples in this topic return the namespace URI for the schema declared in doc.xml:

`http://xsdtesting`

C/C++ Syntax

 Copy

```
HRESULT getDeclaration(IXMLDOMNode* node, ISchemaItem** item);
```

Parameters

`node` [in]

An object. The DOM node.

`item` [out,retval]

An object. The schema object for the specified DOM.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the DOM node points to some object other than a Node or NULL.

E_POINTER

The value returned if the item object is NULL

E_FAIL

The value returned if the node does not have a corresponding declaration in the schema.

ⓘ Note

The syntax `oSchemaCollection.getDeclaration(oDOMNode)` is no longer supported and will return E_NOTIMPL.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMSchemaCollection2](#)-[XMLDOMSchemaCollection](#)

See Also

[IXMLDOMNode](#)

getElementsByTagName Method (DOMDocument)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Returns a collection of elements that have the specified name.

JScript Syntax

 Copy

```
var objXMLDOMNodeList =
    oXMLDOMDocument.getElementsByTagName(tagName);
```

Parameters

`tagName`

A string specifying the element name to find. The `tagName` value `"*"` returns all elements in the document.

Return Value

An object. Points to a collection of elements that match the specified name.

Example

The following script example creates an `IXMLDOMNodeList` object using the `DOMDocument` object's `getElementsByName` method, and then displays all of the elements with the desired tag name.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
var objNodeList;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    objNodeList = xmlDoc.getElementsByTagName("author");
    for (var i=0; i<objNodeList.length; i++) {
        WScript.Echo(objNodeList.item(i).xml);
    }
}
```

When used with the [sample XML file \(books.xml\)](#), this example returns the following output, which includes the full XML string for each instance of the `<author>` element in books.xml:

```
<author>Gambardella, Matthew</author>
```

```
<author>Ralls, Kim</author>
```

```
<author>Corets, Eva</author>
```

```
<author>Corets, Eva</author>
```

```
<author>Corets, Eva</author>
```

```
<author>Randall, Cynthia</author>
```

```
<author>Thurman, Paula</author>
```

```
<author>Knorr, Stefan</author>
```

```
<author>Kress, Peter</author>
```

```
<author>O'Brien, Tim</author>
```

```
<author>O'Brien, Tim</author>
```

<author>Galos, Mike</author>

C/C++ Syntax

 Copy

```
HRESULT getElementsByTagName(
    BSTR tagName,
    IXMLDOMNodeList **resultList);
```

Parameters

`tagName` [in]

The element name to find. The `tagName` value "*" returns all elements in the document.

`resultList` [out, retval]

The address of a collection of elements that match the specified name.

Return Values

`S_OK`

The value returned if successful.

Example

C++

 Copy

```
IXMLDOMDocument *pIXMLDOMDocument = NULL;
wstring strFindText (_T("author"));
IXMLDOMNodeList *pIDOMNodeList = NULL;
IXMLDOMNode *pIDOMNode = NULL;
long value;
BSTR bstrItemText;
HRESULT hr;

try
{
    // Initialize pIXMLDOMDocument (create a DOMDocument).
    // Load document.
    hr = pIXMLDOMDocument->getElementsByName(
        (TCHAR*)strFindText.data(), &pIDOMNodeList);
    SUCCEEDED(hr) ? 0 : throw hr;
```

```

hr = pIDOMNodeList->get_length(&value);
if(SUCCEEDED(hr))
{
    pIDOMNodeList->reset();
    for(int ii = 0; ii < value; ii++)
    {
        pIDOMNodeList->get_item(ii, &pIDOMNode);
        if(pIDOMNode )
        {
            pIDOMNode->get_text(&bstrItemText);
            ::MessageBox(NULL, bstrItemText,strFindText.data(), MB_OK);
            pIDOMNode->Release();
            pIDOMNode = NULL;
        }
    }
}
pIDOMNodeList->Release();
pIDOMNodeList = NULL;
}
catch(...)
{
    if(pIDOMNodeList)
        pIDOMNodeList->Release();
    if(pIDOMNode)
        pIDOMNode->Release();
    DisplayErrorToUser();
}
// Release pIXMLDOMDocument when finished with it.

```

Remarks

The elements in the collection are returned in the order in which they would be encountered in a preorder traversal of the document tree. In a preorder traversal, the parent root node is visited first and each child node from left to right is then traversed.

The returned `IXMLDOMNodeList` object is live and immediately reflects changes to the nodes that appear in the list.

The `getElementsByName` method simulates the matching of the provided argument against the result of the `tagName` property of `IXMLDOMELEMENT`. When executed, it does not recognize or support namespaces. Instead you should use the `selectNodes` method, which is faster in some cases and can support more complex searches.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument-DOMDocument](#)

See Also

[IXMLDOMNodeList](#)

[selectNodes Method](#)

getElementsByTagName Method (IXMLDOMELEMENT)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Returns a list of all descendant elements that match the supplied name.

JScript Syntax

 Copy

```
var objXMLDOMNodeList = oXMLDOMELEMENT.getElementsByTagName(tagName);
```

Parameters

`tagName`

A string specifying the name of the element to find. The `tagName` value `"*"` matches all descendant elements of this element.

Return Value

An object. Returns `IXMLDOMNodeList` object containing all elements that match the supplied name.

Example

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, nodelistAuthor;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodelistAuthor = nodeBook.getElementsByTagName("author");
    WScript.Echo(nodelistAuthor.item(0).text);
}
```

Output

When used with the [sample XML file \(books.xml\)](#), this example returns the following message box output:

Gambardella, Matthew

C/C++ Syntax

```
HRESULT getElementsByTagName(
    BSTR tagName,
    IXMLDOMNodeList **resultList);
```

Parameters

`tagName` [in]

The name of the element to find. The `tagName` value `"*"` matches all descendant elements of this element.

`resultList` [out, retval]

An `IXMLDOMNodeList` object containing all elements that match the supplied name.

Return Values

S_OK

The value returned if successful.

Example

```
c++ Copy
// Find all Nodes with a particular tag.
BOOL DOMDocFindNodesWithTag()
{
    BOOL bResult = FALSE;
    CString strFindText (_T("author"));
    IXMLDOMNodeList *pIDOMNodeList = NULL;
    IXMLDOMNode *pIDOMNode = NULL;
    long value = 0;
    BSTR bstrItemText = NULL;
    HRESULT hr;

    try
    {
        // Create the DOMDocument and initialise m_pIXMLDOMDocument2.
        // Find all "author" elements.
        hr = m_pIXMLDOMDocument2->getElementsByName(
(TCHAR *)strFindText.GetBuffer(0),
&pIDOMNodeList);
        SUCCEEDED(hr) ? 0 : throw hr;

        // Get the length of the list returned by the previous find.
        hr = pIDOMNodeList->get_length(&value);
        if(SUCCEEDED(hr))
        {
            pIDOMNodeList->reset();
            // Loop through the elements found and display the contents.
            for(int ii = 0; ii < value; ii++)
            {
                hr = pIDOMNodeList->get_item(ii, &pIDOMNode);
                SUCCEEDED(hr) ? 0 : throw hr;
                if(pIDOMNode)
                {
                    hr = pIDOMNode->get_text(&bstrItemText);
                    SUCCEEDED(hr) ? 0 : throw hr;
                    if(bstrItemText)
                    {
                        bResult = TRUE;
                        ::MessageBox(NULL, bstrItemText, strFindText, MB_OK);
                        ::SysFreeString(bstrItemText);
                        bstrItemText = NULL;
                    }
                }
            }
        }
    }
}
```

```

        pIDOMNode->Release();
        pIDOMNode = NULL;
    }
}
pIDOMNodeList->Release();
pIDOMNodeList = NULL;
}
}
catch(...)
{
    if(pIDOMNodeList)
        pIDOMNodeList->Release();
    if(pIDOMNode)
        pIDOMNode->Release();
    if(bstrItemText)
        ::SysFreeString(bstrItemText);
    bResult = FALSE;
    DisplayErrorToUser();
}

return bResult;
}

```

Remarks

Elements appear in the order encountered in a preorder traversal of this element's tree.

The `IXMLDOMNodeList` object is returned even if there are no matches. In this case, the length of the list will be set to zero.

The `IXMLDOMNodeList` is live and immediately reflects changes to the nodes that appear in the list.

The `getElementsByTagName` method simulates the matching of the provided argument against the result of the `tagName` property of `IXMLDOMELEMENT`. When executed, it does not recognize or support namespaces. Instead, you should use the `selectNodes` method, which is faster in some cases and can support more complex searches.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMElement](#)

See Also

[IXMLDOMNodeList](#)

getNamedItem Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the attribute with the specified name.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeMap.getNamedItem(name);
```

Parameters

`name`

A string specifying the name of the attribute.

Return Value

An object. Returns `IXMLDOMNode` object for the specified attribute. Returns Nothing if the attribute node is not in this collection.

Example

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, nodeId;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodeId = nodeBook.attributes.getNamedItem("id");
    WScript.Echo(nodeId.value);
}
```

Output

When used with the [sample XML file \(books.xml\)](#), this example returns "bk101", the value of the `id` attribute for the first instance of the `<book>` element in books.xml.

C/C++ Syntax

 Copy

```
HRESULT getNamedItem(
    BSTR name,
    IXMLDOMNode **namedItem);
```

Parameters

name[in]

The name of the attribute.

namedItem[out, retval]

An `IXMLDOMNode` object for the specified attribute. Returns Null if the attribute node is not in this collection.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value when returning Null.

E_INVALIDARG

The value returned if the `namedItem` parameter is Null.

Example

```
c++ Copy
IXMLDOMNode *pIXMLDOMNode = NULL;
IXMLDOMNamedNodeMap *pIXMLDOMNamedNodeMap = NULL;
BSTR bstrAttributeName = ::SysAllocString(_T("dateModified"));
IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;
IXMLDOMDocument *pIXMLDOMDocument = NULL;
VARIANT varValue;

try
{
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.
    // Load/create an XML fragment.
    hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
    SUCCEEDED(hr) ? 0 : throw hr;

    if(pIXMLDOMELEMENT)
    {
        hr = pIXMLDOMELEMENT->get_attributes(&pIXMLDOMNamedNodeMap);
        if(SUCCEEDED(hr) && pIXMLDOMNamedNodeMap)
        {
            hr = pIXMLDOMNamedNodeMap->getNamedItem(bstrAttributeName,
&pIXMLDOMNode);
            if(SUCCEEDED(hr) && pIXMLDOMNode)
            {
                pIXMLDOMNode->get_nodeValue(&varValue);
                ::MessageBox(NULL, _bstr_t(varValue), _T("Item Value"), MB_OK);
                pIXMLDOMNode->Release();
                pIXMLDOMNode = NULL;
            }
            pIXMLDOMNamedNodeMap->Release();
            pIXMLDOMNamedNodeMap = NULL;
        }
        pIXMLDOMELEMENT->Release();
        pIXMLDOMELEMENT = NULL;
    }
    ::SysFreeString(bstrAttributeName);
    bstrAttributeName = NULL;
}
catch(...)
{
    if(bstrAttributeName)
```

```
    ::SysFreeString(bstrAttributeName);
    if(pIXMLDOMELEMENT)
        pIXMLDOMELEMENT->Release();
    if(pIXMLDOMNamedNodeMap)
        pIXMLDOMNamedNodeMap->Release();
    if(pIXMLDOMNode)
        pIXMLDOMNode->Release();
    DisplayErrorToUser();
}
// Release pIXMLDOMDocument when finished with it.
```

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMNodeMap](#)

See Also

[IXMLDOMNode](#)

getOption Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Returns the value of one of the following options:

- `SXH_OPTION_URL` (Read-only.)
- `SXH_OPTION_URL_CODEPAGE`
- `SXH_OPTION_ESCAPE_PERCENT_IN_URL`
- `SXH_OPTION_IGNORE_SERVER_SSL_CERT_ERROR_FLAGS`
- `SXH_OPTION_SELECT_CLIENT_SSL_CERT`

For more information about these options, see Remarks.

JScript Syntax

 Copy

```
varValue = oServerXMLHttpRequest.getOption(option);
```

Parameters

`option`

The option whose value is to be returned.

Example

JavaScript

 Copy

```
var xmlServerHttp = new ActiveXObject("Msxml2.ServerXMLHTTP.3.0");
var optvalue = xmlServerHttp.getOption(2);
WScript.Echo(optvalue);
```

Output

13056

Note

This example returns the current value of the `SXH_OPTION_IGNORE_SERVER_SSL_CERT_ERROR_FLAGS` (option 2), which by default is 13056. This value maps to the `SXH_SERVER_CERT_IGNORE_ALL_SERVER_ERRORS` flag, indicating that the current XMLHTTP server instance will return all certificate errors.

C/C++ Syntax

```
HRESULT getOption(SERVERXMLHTTP_OPTION option, VARIANT * value);
```

Parameters

`option [in]`

The option whose value is to be returned.

`value [out, retval]`

The return value of the option.

Return Values

S_OK

Value returned if successful.

Remarks

The following table lists the options available for the `getOption` method.

Option	Description
-1	<code>SXH_OPTION_URL</code>
	<p>By default, returns a variant of type string that contains the URL of the resource. Read-only.</p> <p>The <code>SXH_OPTION_URL</code> option allows the client to learn the final URL last used. For example, after a call made using the <code>send</code> method has completed, you can use this option to check the URL and see if any redirection occurred. The URL, however, cannot be read until the <code>open</code> method is next called.</p>
0	<code>SXH_OPTION_URL_CODEPAGE</code>
	<p>By default, CP_UTF8 is the code page used to convert the Unicode URL string (specified in the <code>open</code> method) to a single-byte representation.</p> <p>The <code>SXH_OPTION_URL_CODEPAGE</code> option allows the client to override this default with a different code page value. The client should specify an unsigned integer value for the code page.</p>
1	<code>SXH_OPTION_ESCAPE_PERCENT_IN_URL</code>
	<p>By default, escaping unsafe ANSI characters in the URL (for example, " " -> "%20") does not escape the % character itself.</p> <p>The <code>SXH_OPTION_ESCAPE_PERCENT_IN_URL</code> option allows the client to change this behavior. The client should specify a Boolean True/False value for this option.</p>

Option	Description
2	<code>SXH_OPTION_IGNORE_SERVER_SSL_CERT_ERROR_FLAGS</code>
	The <code>SXH_OPTION_IGNORE_SERVER_SSL_CERT_ERROR_FLAGS</code> option is a DWORD mask of various flags that can be set to change this default behavior. The default value is to ignore all problems. You must set this option before calling the <code>send</code> method. The flags are as follows:
	<code>SXH_SERVER_CERT_IGNORE_UNKNOWN_CA = 256</code>
	Unknown certificate authority
	<code>SXH_SERVER_CERT_IGNORE_WRONG_USAGE = 512</code>
	Malformed certificate such as a certificate with no subject name.
	<code>SXH_SERVER_CERT_IGNORE_CERT_CN_INVALID = 4096</code>
	Mismatch between the visited hostname and the certificate name being used on the server.
	<code>SXH_SERVER_CERT_IGNORE_CERT_DATE_INVALID = 8192</code>
	The date in the certificate is invalid or has expired.
	<code>SXH_SERVER_CERT_IGNORE_ALL_SERVER_ERRORS = 13056</code>
	All certificate errors.
	To turn off a flag, you subtract it from the default value, which is the sum of all flags. For example, to catch an invalid date in a certificate, you turn off the <code>SXH_SERVER_CERT_IGNORE_CERT_DATE_INVALID</code> flag as follows:
	<pre>shx.setOption(2) = (shx.getOption(2) - SXH_SERVER_CERT_IGNORE_CERT_DATE_INVALID)</pre>

Option	Description
3	SXH_OPTION_SELECT_CLIENT_SSL_CERT By default, the value of this option is an empty string (""), which means pick the first certificate in the local store to send if the server requests a client certificate. The <code>SXH_OPTION_SELECT_CLIENT_SSL_CERT</code> option is a string that lets you select which client certificate from the local store should be sent. You must set this option before calling the <code>send</code> method. The following example sets the client certificate option to request the client certificate named "MSXML": <code>certName = shx.getOption(3) = "MSXML"</code>

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IServerXMLHTTPRequest](#)-[ServerXMLHTTP](#)

See Also

[open Method \(ServerXMLHTTP-I ServerXMLHTTPRequest\)](#)
[setOption Method](#)

getProperty Method (IXMLDOMDocument2)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the value of one of the [second-level properties](#) that are set either by default or using the [setProperty](#) method.

JScript Syntax

jscript

 Copy

```
strPropValue = objXMLDOMDocument2.getProperty(name);
```

Parameters

name

The string name of the property. This name is case-sensitive.

Example

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
xmlDoc.setProperty("SelectionLanguage", "XPath");
WScript.Echo(xmlDoc.getProperty("SelectionLanguage"));
```

Output

XPath

C/C++ Syntax

 Copy

```
HRESULT getProperty(BSTR name, VARIANT* value);
```

Parameters

`name` [in]

The string name of the property. This name is case-sensitive.

`value` [out, retval]

The variant return value of the requested flag.

Return Values

S_OK

The value returned if successful.

E_FAIL

The value returned if property name is invalid.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMDocument2](#)

See Also

Second-Level DOM Properties

setProperty Method1

getProperty Method (IXMLDOMSelection)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Returns a property.

JScript Syntax

 Copy

```
strValue = objXMLDOMSelection.getProperty(name);
```

Parameters

name

The string name of the property. This name is case-sensitive.

Example

 Jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var oSelection;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
```

```
WScript.Echo("You have error " + myErr.reason);
} else {
    oSelection = xmlDoc.selectNodes("//book");
    WScript.Echo(oSelection.getProperty("SelectionLanguage"));
}
```

Output

 Copy

XPath

C/C++ Syntax

 Copy

```
HRESULT getProperty(BSTR name, VARIANT* value);
```

Parameters

`name` [in]

The string name of the property. This name is case-sensitive.

`value` [out, retval]

The variant return value of the requested property.

Return Values

`S_OK`

The value returned if method successful.

`E_INVALIDARG`

The value returned if named property does not exist.

Remarks

The `getProperty` method returns the value for the internal `SelectionLanguage` property (flag) that was set by calling the `setProperty` method on the document or the default

value.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMSelection](#)

See Also

[setProperty Method1](#)

getQualifiedItem Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Visual Basic Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

Returns the attribute with the specified namespace and attribute name.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeMap.getQualifiedItem(baseName,  
namespaceURI);
```

Parameters

`baseName`

The string specifying the base name of the attribute, without namespace qualification.

`namespaceURI`

The string specifying the namespace prefix that qualifies the attribute name.

Return Value

An object. Returns the attribute node specified by the `baseName` and `namespaceURI` parameters. Returns Null if the attribute is not in the collection or if the item is not an attribute.

Example

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var oNamedNodeMap, nodeBook;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    oNamedNodeMap = nodeBook.attributes;
    WScript.Echo(oNamedNodeMap.getQualifiedItem("id", "").value);
}
```

Output

When used with the [sample XML file \(books.xml\)](#), this example returns "bk101", the value of the `id` attribute for the first instance of the `<book>` element in `books.xml`.

Visual Basic Syntax

 Copy

```
Set objXMLDOMNode = oXMLDOMNamedNodeMap.getQualifiedItem  
(baseName, namespaceURI)
```

Parameters

`baseName`

The string specifying the base name of the attribute, without namespace qualification.

`namespaceURI`

The string specifying the namespace prefix that qualifies the attribute name.

Return Value

An object. Returns the attribute node specified by the `baseName` and `namespaceURI` parameters. Returns Null if the attribute is not in the collection or if the item is not an attribute.

Output

When used with the [sample XML file \(books.xml\)](#), this example returns "bk101", the value of the `id` attribute for the first instance of the `<book>` element in `books.xml`.

C/C++ Syntax

 Copy

```
HRESULT getQualifiedItem(
    BSTR baseName,
    BSTR namespaceURI,
    IXMLDOMNode **qualifiedItem);
```

Parameters

`baseName` [in]

The base name of the attribute, without namespace qualification.

`namespaceURI` [in]

The namespace prefix that qualifies the attribute name.

`qualifiedItem` [out, retval]

The attribute node specified by the `baseName` and `namespaceURI` parameters. Returns Null if the attribute is not in the collection or if the item is not an attribute.

Return Values

`S_OK`

The value returned if successful.

`S_FALSE`

The value when returning Null.

E_INVALIDARG

The value returned if `qualifiedItem` is Null.

Example

```
C++ Copy

IXMLDOMNode *pIXMLDOMNode = NULL;
IXMLDOMNodeMap *pIXMLDOMNodeMap = NULL;
BSTR bstrAttributeName = ::SysAllocString(_T("dateModified"));
IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;
VARIANT varValue;
IXMLDOMDocument *pIXMLDOMDocument = NULL;
HRESULT hr;

try
{
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.
    // Load/create an XML fragment.
    hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
    SUCCEEDED(hr) ? 0 : throw hr;

    if(pIXMLDOMELEMENT)
    {
        hr = pIXMLDOMELEMENT->get_attributes(&pIXMLDOMNodeMap);
        if(SUCCEEDED(hr) && pIXMLDOMNodeMap)
        {
            hr = pIXMLDOMNodeMap->getQualifiedItem(bstrAttributeName, NULL,
&pIXMLDOMNode);
            if(SUCCEEDED(hr) && pIXMLDOMNode)
            {
                pIXMLDOMNode->get_nodeValue(&varValue);
                ::MessageBox(NULL, _bstr_t(varValue), _T("Item Value"), MB_OK);
                pIXMLDOMNode->Release();
                pIXMLDOMNode = NULL;
            }
            pIXMLDOMNodeMap->Release();
            pIXMLDOMNodeMap = NULL;
        }
        pIXMLDOMELEMENT->Release();
        pIXMLDOMELEMENT = NULL;
    }
    ::SysFreeString(bstrAttributeName);
    bstrAttributeName = NULL;
}
catch(...)
{
    if(bstrAttributeName)
        ::SysFreeString(bstrAttributeName);
    if(pIXMLDOMELEMENT)
        pIXMLDOMELEMENT->Release();
```

```
if(pIXMLDOMNodeMap)
    pIXMLDOMNodeMap->Release();
if(pIXMLDOMNode)
    pIXMLDOMNode->Release();
DisplayErrorToUser();
}
// Release pIXMLDOMDocument when finished with it.
```

Remarks

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMNodeMap](#)

getResponseHeader Method (IXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the value of an HTTP header from the response body.

JScript Syntax

 Copy

```
strValue = oXMLHttpRequest.getResponseHeader(bstrHeader);
```

Parameters

`bstrHeader`

A string containing the case-insensitive header name.

Return Value

A string. Contains the resulting header information.

Example

jscript

 Copy

```
var xmlhttp = new ActiveXObject("MSXML2.XMLHTTP.6.0");
xmlhttp.open("GET", "http://localhost/sample.xml", false);
xmlhttp.send();
WScript.Echo(xmlhttp.getResponseHeader("Server"));
```

Output

Returns the value of the Server field in the HTTP header, which indicates the current version of the Web server running locally. In this case, the value is "Microsoft-IIS/5.1" for Internet Information Services (IIS) version 5.1, which is the version provided if the local computer is running under Microsoft Windows XP.

C/C++ Syntax

 Copy

```
HRESULT getResponseHeader(BSTR bstrHeader, BSTR* pbstrValue);
```

Parameters

bstrHeader [in]

A case-insensitive header name.

pbstrValue [out, retval]

The resulting header information.

Return Values

S_OK

The value returned if successful.

Example

 Copy

```
c++
HRESULT hr;
BSTR bstrValue = NULL;
IXMLHttpRequest *pIXMLHttpRequest = NULL;

try
```

```
{  
    // Create XMLHttpRequest object and initialize pIXMLHttpRequest.  
    hr = pIXMLHttpRequest->getResponseHeader(_T("Server"), &bstrValue);  
    if(SUCCEEDED(hr))  
    {  
        ::MessageBox(NULL, m_bstrValue, _T("Response Header-Server"), MB_OK);  
        ::SysFreeString(bstrValue);  
        bstrValue = NULL;  
    }  
}  
catch(...)  
{  
    if(bstrValue)  
        ::SysFreeString(bstrValue);  
    DisplayErrorToUser();  
}  
// Release pIXMLHttpRequest when finished with it.
```

Remarks

The results of this method are valid only after the `send` method has been successfully completed. The line, `xmlhttp.getResponseHeader("Content-Type");`, returns the string `"text/xml"`, assuming the server set `"text/xml"` as the content type. The full list of header variables you can query can be accessed from the `getAllResponseHeaders` method.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLHTTPRequest](#)

See Also

[getAllResponseHeaders Method \(IXMLHTTPRequest\)](#)

[send Method \(IXMLHTTPRequest\)](#)

[setRequestHeader Method \(IXMLHTTPRequest\)](#)

getResponseHeader Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the value of an HTTP header from the response body.

JScript Syntax

 Copy

```
strValue = oServerXMLHTTPRequest.getResponseHeader(bstrHeader);
```

Parameters

`bstrHeader`

A case-insensitive header name.

Example

JavaScript

 Copy

```
var xmlServerHttp = new ActiveXObject("Msxml2.ServerXMLHTTP.6.0");
xmlServerHttp.open("GET", "http://localhost/sample.xml", false);
```

```
xmlServerHttp.send();
WScript.Echo(xmlServerHttp.getResponseHeader("Server"));
```

Output

Returns the value of the Server field in the HTTP header, which indicates the current version of the Web server running locally. In this case, the value is "Microsoft-IIS/5.1" for Internet Information Services (IIS) version 5.1, which is the version provided if the local computer is running under Microsoft Windows XP.

C/C++ Syntax

 Copy

```
HRESULT getResponseHeader(BSTR bstrHeader, BSTR* pbstrValue);
```

Parameters

`bstrHeader` [in]

A case-insensitive header name.

`pbstrValue` [out, retval]

The resulting header information.

Return Values

`S_OK`

The value returned if successful.

Remarks

The results of this method are valid only after the `send` method has been successfully completed. The line, `oServerXMLHttpRequest.getResponseHeader("Content-Type");` returns the string "`text/xml`", assuming the server set "`text/xml`" as the content type.

The full list of header variables you can query can be accessed from the `getAllResponseHeaders` method.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IServerXMLHTTPRequest](#)-[ServerXMLHTTP](#)

See Also

[getAllResponseHeaders Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[send Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[setRequestHeader Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

GetResponseHeader Method (IXMLHttpRequest2, Windows 8)

10/27/2016 • 2 minutes to read

In this article

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

Retrieves the value of an HTTP header from the HTTP response headers.

C/C++ Syntax

 Copy

```
HRESULT GetResponseHeader(const WCHAR *pwszHeader, WCHAR *ppwszValue);
```

Parameters

pwszHeader [in, string, ref]

A case-insensitive header name.

ppwszValue [out, string]

The resulting header information. You should free the memory for this parameter by calling the `CoTaskMemFree` function.

Return Values

S_OK

The value returned if successful.

Example

```
c++ Copy
HRESULT hr;
const WCHAR pwszHeaderValue = NULL;
IXMLHttpRequest2 *pIXMLHttpRequest2 = NULL;

// Create XMLHttpRequest2 object and initialize pIXMLHTTP2Request.
hr = pIXMLHttpRequest2->getResponseHeader(L"Server"), &pwszHeaderValue);
if(SUCCEEDED(hr))
{
    ::MessageBox(NULL, m_pwszHeaderValue, L"Response Header-Server", MB_OK);
    ::SysFreeString(pwszHeaderValue);
    pwszHeaderValue = NULL;
}

// Release pIXMLHttpRequest2 when finished with it.
```

Remarks

The results of this method are valid only after the `OnHeadersAvailable` callback has been executed. The line `xmlhttp.getResponseHeader("Content-Type");` returns the string `"text/xml"`, assuming the server set `"text/xml"` as the content type. Call the `GetAllResponseHeaders` method to receive the full list of header variables you can query.

Versioning

Implemented in: MSXML 6.0

Applies to

[IXMLHTTPRequest2 \(Windows 8\)](#)

See Also

[GetAllResponseHeaders Method \(IXMLHTTPRequest2, Windows 8\)](#)

[Send Method \(IXMLHTTPRequest2, Windows 8\)](#)

[SetRequestHeader Method \(IXMLHTTPRequest2, Windows 8\)](#)

[CoTaskMemFree Function](#)

getSchema Method1

05/17/2017 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[Applies to](#)

[See Also](#)

[This feature was implemented only for MSXML 6.0.]

Returns an `ISchema` object.

In MSXML 6.0, if you add a schema to the schema cache that imports a schema with a different target namespace, and you call this method, you will get only the objects from the specified target namespace. If you want to get the `ISchema` object populated with the imported namespace schema, you must call this method, passing the imported namespace.

JScript Syntax

 Copy

```
var oSchema = oSchemaCache.getSchema(strNamespaceURI);
```

Parameters

`strNamespaceURI`

A string. The namespace of the schema to be retrieved from the schema cache.

Return Values

oSchema

An object. The schema object for the Namespace URI sent to the `getSchema` method.

Example

The following JScript example shows the `getSchema` method being used to return a schema object.

jscript

 Copy

```
var oSchemaCache = new ActiveXObject("Msxml2.XMLSchemaCache.6.0");
var nsTarget = "http://xsdtesting";
oSchemaCache.add(nsTarget, "doc.xsd");
var oSchema = oSchemaCache.getSchema(nsTarget);
WScript.Echo(oSchema.targetNamespace);
```

C/C++ Syntax

 Copy

```
HRESULT getSchema(BSTR namespaceURI, ISchema** schema);
```

Parameters

strNamespaceURI [in]

A string. The namespace of the schema to be retrieved from the schema cache.

oSchema [out,retval]

An object. The schema object for the specified schema.

Return Values

S_OK

The value returned if successful.

E_POINTER

The value returned if the schema is NULL

E_FAIL

The value returned if another problem is detected.

Versioning

Implemented in: MSXML 6.0

Applies to

[IXMLDOMSchemaCollection2](#)-[XMLDOMSchemaCollection](#)

See Also

[ISchema Interface](#)

[Using Namespaces in Schemas](#)

hasChildNodes Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Provides a fast way to determine whether a node has children.

JScript Syntax

 Copy

```
boolValue = oXMLDOMNode.hasChildNodes();
```

Return Value

Boolean. Returns True if this node has children.

Example

The following script example checks a node to see if it has any child nodes. If it does, it displays the number of child nodes it contains.

Note

You can use **books.xml** to run this sample code.

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var currNode;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    currNode = xmlDoc.documentElement.firstChild;
    if (currNode.hasChildNodes()) {
        WScript.Echo(currNode.childNodes.length);
    } else {
        WScript.Echo("no child nodes");
    }
}
```

Output

 Copy

6

C/C++ Syntax

 Copy

```
HRESULT hasChildNodes(
    VARIANT_BOOL *hasChild);
```

Parameters

`hasChild` [out, retval]

Returns True if this node has children.

Return Values

`S_OK`

The value returned if successful.

`S_FALSE`

The value returned when there are no children.

E_INVALIDARG

The value returned if the `hasChild` parameter is Null.

Remarks

It always returns False for nodes that, by definition, cannot have children: the `IXMLDOMCDATASection`, `IXMLDOMComment`, `IXMLDOMNodeotation`, `IXMLDOMProcessingInstruction`, and `IXMLDOMText` nodes.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMAttribute](#)
[IXMLDOMCDATASection](#)
[IXMLDOMCharacterData](#)
[IXMLDOMComment](#)
[IXMLDOMDocument-DOMDocument](#)
[IXMLDOMDocumentFragment](#)
[IXMLDOMDocumentType](#)
[IXMLDOMElement](#)
[IXMLDOMEntity](#)
[IXMLDOMEntityReference](#)
[IXMLDOMNode](#)
[IXMLDOMNodeotation](#)
[IXMLDOMProcessingInstruction](#)
[IXMLDOMText](#)

hasFeature Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Indicates support for the specified feature.

JScript Syntax

 Copy

```
boolVal = objXMLDOMImplementation.hasFeature(feature, version);
```

Parameters

feature

A string that specifies the feature to test. In Level 1, valid feature values are "XML", "DOM", and "MS-DOM" (case-insensitive).

version

A string that specifies the version number to test, or if Null, tests for implementation of the feature in any version. In Level 1, "1.0" is the valid version value.

Return Value

Boolean. Returns True if the specified feature is implemented; otherwise False.

Example

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var bool = xmlDoc.implementation.hasFeature("DOM", "1.0");
WScript.Echo(bool);
```

Output

-1

C/C++ Syntax

```
HRESULT hasFeature(
    BSTR feature,
    BSTR version,
    VARIANT_BOOL *hasFeature);
```

 Copy

Parameters

feature [in]

The feature to test. In Level 1, valid feature values are "XML", "DOM", and "MS-DOM" (case-insensitive).

version [in]

The version number to test, or, if Null, tests for implementation of the feature in any version. In Level 1, "1.0" is the only valid version value.

hasFeature [out, retval]

True if the specified feature is implemented; otherwise False.

Return Values

S_OK

The value returned if successful.

Example

C++

Copy

```
IXMLDOMImplementation *pIXMLDOMImplementation = NULL;
VARIANT_BOOL varbFlag ;
BSTR bstrOutput = NULL;
BSTR bstrFeature = ::SysAllocString(_T("MS-DOM"));
HRESULT hr;
IXMLDOMDocument *pIXMLDOMDocument = NULL;

try
{
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.
    // Load/create an XML fragment.
    hr = pIXMLDOMDocument->getImplementation(&pIXMLDOMImplementation);

    if(SUCCEEDED(hr) && pIXMLDOMImplementation)
    {
        hr = pIXMLDOMImplementation->hasFeature(bstrFeature, _T("1.0"),
&varbFlag);
        if(varbFlag == VARIANT_TRUE )
            bstrOutput = ::SysAllocString(_T("Feature Supported"));
        else
            bstrOutput = ::SysAllocString(_T("Feature not Supported"));

        ::MessageBox(NULL, bstrOutput, bstrFeature, MB_OK);
        ::SysFreeString(bstrOutput);
        bstrOutput = NULL;
        ::SysFreeString(bstrFeature);
        bstrFeature = NULL;
        pIXMLDOMImplementation->Release();
    }
}
catch(...)
{
    if(bstrOutput)
        ::SysFreeString(bstrOutput);
    if(bstrFeature)
        ::SysFreeString(bstrFeature);
    if(pIXMLDOMImplementation)
        pIXMLDOMImplementation->Release();
    DisplayErrorToUser();
}
// Release pIXMLDOMDocument when finished using it.
```

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLODImplementation](#)

importNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Visual Basic Syntax](#)

[C/C++ Syntax Using Smart Pointers](#)

[Remarks](#)

[Example](#)

[Applies to](#)

[Versioning](#)

[See Also](#)

[This sample code uses features that were implemented only in MSXML 6.0.]

This method can be used to clone a node from a DOM object. In addition to supporting all the functionality offered by the `cloneNode` method, it enables cloning a node from DOM objects of different threading models. The clones created this way can be passed between such DOM objects. For example, a node from a free-threaded DOM object can be passed into an apartment-threaded DOM object. The resultant node object can then be added into the document using the `appendChild` method.

Note

In MSXML, "free-threaded" means `ThreadingModel='Both'`, and cross-thread marshalling is supported.

JScript Syntax

 Copy

```
var clone = objXMLDOMDocument6.importNode(node, deep);
```

Visual Basic Syntax

 Copy

```
Set clone = objXMLDOMDocument6.importNode(node, deep);
```

C/C++ Syntax Using Smart Pointers

 Copy

```
IXMLDOMNodePtr clone = objXMLDOMDocument6->importNode(node, deep);
HRESULT IXMLDOMDocument3::importNode(
...[in] IXMLDOMNode* node,
[in] VARIANT_BOOL deep,
...[out, retval] IXMLDOMNode** clone);
```

Parameters

`node`

An `IXMLDOMNode` object to be cloned.

`deep`

If `true`, any children of `node` will be cloned as well. If `false`, no children of `node` will be cloned.

`clone`

A clone of the `node` object, as specified above.

Return Values

`S_OK`

The node is cloned successfully.

`E_FAIL`

The node cannot be cloned properly and the resultant `clone` parameter is NULL.

Remarks

The `importNode` method does not allow for import of DOM nodes that are of one of the following enumerated node types: NODE_DOCUMENT, NODE_DOCUMENT_TYPE, NODE_ENTITY, and NODE_NOTATION.

Example

This example demonstrates how to use `importNode` to clone a node from a free-threaded DOM document and append the clone to a regular apartment-threaded DOM object. As a comparison, it also shows that a DOM object can clone a node from itself with `importNode`, just as it would with `cloneNode`.

We've provided source files for the sample in three languages: JScript, Visual Basic, and C++. The output is the same in each language.

- [Resource Files \(doc1.xml and doc2.xml\)](#)
- [JScript Code \(importNode.js\)](#)
- [C/C++ Code \(importNode.cpp\)](#)
- [Output for importNode Example](#)

Applies to

[IXMLDOMDocument3](#)

Versioning

Implemented in: MSXML 6.0

See Also

[IXMLDOMNode](#)
[IXMLDOMNodeList](#)
[IXMLDOMParseError](#)
[IXMLDOMSchemaCollection-XMLSchemaCache](#)

insertBefore Method

10/27/2016 • 4 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Inserts a child node to the left of the specified node, or at the end of the list.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNode.insertBefore(newChild, refChild);
```

Parameters

`newChild`

An object. The address of the new node to be inserted.

`refChild`

A variant. The address of the reference node; the `newChild` parameter is inserted to the left of the `refChild` parameter. If Null, the `newChild` parameter is inserted at the end of the child list.

Return Value

Object. On success, returns the child node that was inserted.

Example

The following script example creates a new `IXMLDOMNode` object and inserts it as the second child of the top-level node.

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var newNode;
var currNode;
xmlDoc.async = false;
xmlDoc.loadXML("<root><child1/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    WScript.Echo(root.xml);
    newNode = xmlDoc.createNode(1, "CHILD2", "");
    currNode = root.insertBefore(newNode, root.childNodes.item(1));
    WScript.Echo(root.xml);
}
```

C/C++ Syntax

 Copy

```
HRESULT insertBefore(
    IXMLDOMNode *newChild,
    VARIANT refChild,
    IXMLDOMNode **outNewChild);
```

Parameters

`newChild` [in]

The address of the new node to be inserted. The node passed here must be a valid child of the current XML DOM document node. For example, if the current node is an attribute, you cannot pass another attribute in the `newChild` parameter, because an attribute cannot have an attribute as a child.

If `newChild` is of DOCUMENT_FRAGMENT node type, all its children are inserted in order before `refChild`. If `newChild` is already in the tree, it is first removed before it is reinserted before the `refChild` node. Read-only nodes, such as NODE_DOCUMENT_TYPE and NODE_ENTITY nodes, cannot be passed in the `newChild` parameter.

`refChild`[in]

The address of the reference node. The node specified is where the `newChild` node is to be inserted to the left as the preceding sibling in the child list. The node passed here must be either a child node of the current node or Null. If the value is Null, the `newChild` node is inserted at the end of the child list. If the `refChild` node is not a child of the current node, an error is returned.

`outNewChild`[out, retval]

On success, the child node that was inserted. If Null, no object is created.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `newChild` parameter is Null.

E_FAIL

The value returned if an error occurs.

Remarks

The following are general remarks about the `insertBefore` method when DTD/schema rules or namespace prefixes are involved.

How DTD/Schema definitions are resolved

The `newChild` and `refChild` parameters can represent nodes in the same document or in different documents. When in the same document, the nodes retain their default attributes and data types. When in different documents, the nodes either lose or alter their default attributes, depending on whether the document that contains them has a document type definition (DTD) or other schema. MSXML attempts to correctly merge the different DTDs.

If the `newChild` node's DTD or schema differs from that of the original document, the nodes will be treated with the definitions in the new DTD, including default attributes and data types. If there is no DTD, the nodes keep their data types by picking up an instance definition in which elements are defined based upon how they are used in the current DOM document instance but any attributes within the `newChild` node fragment lose their

data types and defaults. Cutting and pasting between documents with two different DTDs can result in an invalid document that might fail to parse after being saved.

How conflicts between prefixes and namespaces are resolved

If there is a conflict between prefixes on the containing element and the prefixes being added, the `insertBefore` operation fails and returns an error. For example, a conflict occurs when a new attribute referring to a different namespace is added to an element with the namespace, as in the following.

 Copy

```
xmlns:myns="URN1"
```

The error can result from the new attribute, where `myns` refers to a different namespace, "URN2," such as would result from a call to

```
createNode("attribute", "myns:myname", "URN2"):
```

 Copy

```
myns:myname="myattributevalue"
```

Namespace conflict errors occur only when setting attributes. Inserted child elements do not cause a namespace conflict.

When adding a document fragment, the containing element adds all namespaces and prefixes from the document fragment. If this causes a conflict with the containing element, `insertBefore` returns an error.

Further remarks

The use and operation of the `insertBefore` method depends on the value of the `nodeType` property for the `IXMLDOMNode` type object passed in the `newChild` parameter. The following topics provide further information about the results of calling `insertBefore` on different types of nodes.

- [Calling insertBefore on Attributes](#)
- [Calling insertBefore on Documents](#)
- [Calling insertBefore on Document Fragments](#)

- Calling insertBefore on Elements

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[createNode Method](#)
[nodeType Property](#)
[IXMLDOMAttribute](#)
[IXMLDOMCDATASection](#)
[IXMLDOMCharacterData](#)
[IXMLDOMComment](#)
[IXMLDOMDocument-DOMDocument](#)
[IXMLDOMDocumentFragment](#)
[IXMLDOMDocumentType](#)
[IXMLDOLElement](#)
[IXMLDOMEntity](#)
[IXMLDOMEntityReference](#)
[IXMLDOMNode](#)
[IXMLDOMNotation](#)
[IXMLDOMProcessingInstruction](#)
[IXMLDOMText](#)

insertData Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Inserts a string at the specified offset.

JScript Syntax

 Copy

```
oXMLDOMCharacterData.insertData(offset, data);
```

Parameters

`offset`

A long integer specifying the offset, in characters, at which to insert the supplied string data.

`data`

A string containing the data that is to be inserted into the existing string.

Example

The following script example creates an `IXMLDOMComment` object and uses the `insertData` method to insert a string into it.

 Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var comment;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    comment = xmlDoc.createComment("Hello World!");
    comment.insertData(6, "Beautiful ");
    WScript.Echo(comment.xml);
}
```

Output

```
<!--Hello Beautiful World!>
```

C/C++ Syntax

```
HRESULT insertData(
    long offset,
    BSTR data);
```

 Copy

Parameters

`offset` [in]

The offset, in characters, at which to insert the supplied string data.

`data` [in]

The string data that is to be inserted into the existing string.

Return Values

`S_OK`

The value returned if successful.

S_FALSE

The value when returning Null.

E_FAIL

The value returned if an error occurs.

Remarks

The `length` property is updated by this operation.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[length Property \(IXMLDOMCharacterData\)](#)

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMText](#)

item Method (IXMLDOMNodeList)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Allows random access to individual nodes within the collection.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeList.item(index);
```

Parameters

index

A long integer. An index of the item within the collection. The first item is zero.

Return Value

An object. Returns `IXMLDOMNode`. Returns Null if the index is out of range.

Example

The following script example creates an `IXMLDOMNodeList` object with the document's `getElementsByName` method. It then iterates through the collection, displaying the text value of each item in the list.

ⓘ Note

You can use **books.xml** to run this sample code.

XML

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var objNodeList;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    objNodeList = xmlDoc.getElementsByTagName("author");
    for (var i=0; i<objNodeList.length; i++) {
        WScript.Echo(objNodeList.item(i).text);
    }
}
```

Output

Gambardella, Matthew

Ralls, Kim

Corets, Eva

...

C/C++ Syntax

```
HRESULT get_item(
    long index,
    IXMLDOMNode **listItem);
```

Parameters

index [in]

An index of the item within the collection. The first item is number zero.

`listItem[out, retval]`

An `IXMLDOMNode`. Returns Null if the index is out of range.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `listItem` parameter is Null.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNode](#)

[IXMLDOMNodeList](#)

[IXMLDOMSelection](#)

item Method (IXMLDOMNodeNamedNodeMap)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Allows random access to individual nodes within the collection.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeNamedNodeMap.item(index);
```

Parameters

`index`

A long integer. An index of the item within the collection. The first item is zero.

Return Value

An object. Returns `IXMLDOMNode`. Returns Null if the index is out of range.

Example

The following script example creates an `IXMLDOMNodeNamedNodeMap` object to retrieve the attributes for an element node selected using the `SelectSingleNode` method. It then

iterates through the attributes, before displaying the name and value of each attribute in the collection.

ⓘ Note

You can use **books.xml** to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var oNamedNodeMap, nodeBook, str;
var str = "";
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xmlDoc.setProperty("SelectionLanguage", "XPath");
    nodeBook = xmlDoc.selectSingleNode("//book");
    oNamedNodeMap = nodeBook.attributes;
    for (var i=0; i<oNamedNodeMap.length; i++) {
        str += "Attr" + i + " name: " +
            oNamedNodeMap.item(i).name + "\n" +
            "Attr" + i + " value: " +
            oNamedNodeMap.item(i).text + "\n";
    }
    WScript.Echo(str);
}
```

Output

Attr0 name: id

Attr0 value: bk101

C/C++ Syntax

 Copy

```
HRESULT get_item(
    long index,
    IXMLDOMNode **listItem);
```

Parameters

`index [in]`

The index of the item within the collection. The first item is zero.

`listItem [out, retval]`

The `IXMLDOMNode` object. Returns Null if the index is out of range.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `listItem` parameter is Null.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNode](#)

[IXMLDOMNodeMap](#)

[IXMLDOMSelection](#)

load Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Loads an XML document from the specified location.

JScript Syntax

 Copy

```
boolValue = oXMLDOMDocument.load(xmlSource);
```

Parameters

xmlSource

A string containing a URL that specifies the location of the XML file.

Return Value

Boolean. Returns True if the load succeeded; False if the load failed.

Example

The following script example creates a `DOMDocument` object and uses the `load` method to load a local XML file.

ⓘ Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    WScript.Echo(xmlDoc.xml);
}
```

Output

Outputs the `books.xml` file.

C/C++ Syntax

```
HRESULT load(
    VARIANT xmlSource,
    VARIANT_BOOL *isSuccessful);
```

Parameters

`xmlSource` [in]

An indicator of the source XML to parse. This may be an URL (String/BSTR), a Request object (in an ASP page), an `IStream`, SAFEARRAY of bytes (VT_ARRAY|VT_UI1), a `DOMDocument` object, or any object that supports `IStream`, `ISequentialStream`, or `IPersistStream`. See Remarks for more information.

`isSuccessful` [out, retval]

True if the load succeeded; False if the load failed.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value returned if the load fails.

E_INVALIDARG

The value returned if the `isSuccessful` parameter is Null.

Remarks

If the URL cannot be resolved or accessed or does not reference an XML document, this method returns an error and sets the `documentElement` property of the `DOMDocument` to Null.

The `load` method can also take any object that supports `IStream` and the Microsoft® Internet Information Services (IIS) `Request` object.

Calling `load` or `loadXML` on an existing document immediately discards the content of the document.

If loading an XML document from a resource, the load must be performed asynchronously or the load will fail. For example:

 Copy

```
Set objXML = CreateObject("Msxml2.DOMDocument.6.0")
ObjXML.async=true
objXML.load "res://msxml3.dll/xml/defaultss.xsl"
If objXML.parseError.errorCode <> 0 Then
    Set myErr = objXML.parseError
    WScript.Echo "You have error " + myErr.reason
Else
    WScript.Echo objXML.parseError.reason
    objXML.save "defaultss.xsl"
End If
```

You can use this method to check if the loaded XML document is well-formed. You cannot use it to validate the XML document against a schema.

This member is an extension of the World Wide Web Consortium (W3C) Document Object Model (DOM).

When a document is loaded using this method, URLs will be resolved relative to the directory from which the program executed.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[documentElement Property](#)

[loadXML Method1](#)

[Persistence and the DOM](#)

[IXMLDOMDocument-DOMDocument](#)

loadXML Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Loads an XML document using the supplied string.

JScript Syntax

 Copy

```
boolValue = oXMLEDOMDocument.loadXML(bstrXML);
```

Parameters

`bstrXML`

A string containing the XML string to load into this XML document object. This string can contain an entire XML document or a well-formed fragment.

Return Value

Boolean. Returns True if the XML load succeeded. Returns False and sets the `documentElement` property of the `DOMDocument` to Null if the XML load failed.

Example

The following script example creates a `DOMDocument` object, and then uses its `loadXML` method to load the specified XML before displaying it.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
xmlDoc.async = false;
xmlDoc.loadXML("<customer><first_name>Joe</first_name>
<last_name>Smith</last_name></customer>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    WScript.Echo(xmlDoc.xml);
}
```

Output

```
<customer><first_name>Joe</first_name><last_name>Smith</last_name></customer>
```

C/C++ Syntax

```
HRESULT loadXML(
    BSTR bstrXML,
    VARIANT_BOOL * isSuccessfull);
```

Parameters

bstrXML [in]

An XML string to load into this XML document object. This string can contain an entire XML document or a well-formed fragment.

isSuccessfull [out, retval]

True if the XML load succeeded. If the XML load failed, this method returns False and sets the `documentElement` property of the `DOMDocument` object to Null.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value returned if the load fails.

E_INVALIDARG

The value returned if the `isSuccessful` parameter is Null.

Remarks

Calling `load` or `loadXML` on an existing document immediately discards the content of the document. The `loadXML()` method will work only with UTF-16 or UCS-2 encodings.

You can use this method to check if the loaded XML document is well-formed. You cannot use it to validate the XML document against a schema.

This member is an extension of the World Wide Web Consortium (W3C) Document Object Model (DOM).

When a document is loaded using this method, URLs will be resolved relative to the directory from which the program executed.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[documentElement Property](#)

[load Method1](#)

[Persistence and the DOM](#)

[IXMLDOMDocument-DOMDocument](#)

matches Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Checks if the node that is passed is contained in the current collection.

JScript Syntax

 Copy

```
objXMLDOMNode = objXMLDOMSelection.matches(objXMLDOMNode);
```

Example

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var oSelection, nodeBook, node;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//child");
    oSelection = xmlDoc.selectNodes("//*");
    node = oSelection.matches(nodeBook);
    WScript.Echo(node.xml);
}
```

Output

Outputs the books.xml file.

C/C++ Syntax

 Copy

```
HRESULT matches(IXMLDOMNode* pNode, IXMLDOMNode** ppNode);
```

Parameters

`pNode` [in]

The node that is passed in.

`ppNode` [out, retval]

The Boolean result.

Return Values

`S_OK`

The value returned if method successful.

`E_INVALIDARG`

The value returned if the `pNode` parameter isNull.

An error for cases that are not allowed, as outlined in the Extensible Stylesheet Language (XSL) specification (applied to ID, IDREF, and Ancestor).

Remarks

When `matches` is called with node A, it returns node B such that if B was set as the context on a query, A is in the result set of the query. If no such B node is found, `matches` returns Null.

The `matches` method does not take into account the current context of the query.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMSelection](#)

nextNode Method (IXMLDOMNodeList)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Returns the next node in the collection.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeList.nextNode();
```

Return Value

An `IXMLDOMNode` refers to the next node in the collection. Returns Null if there is no next node.

Example

The following script example creates an `IXMLDOMNodeList` object and uses its `nextNode` method to iterate the collection.

Note

You can use `books.xml` to run this sample code.

 Copy

```
JavaScript
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var objNodeList;
var objNode;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    objNodeList = xmlDoc.getElementsByTagName("author");
    for (var i=0; i<objNodeList.length; i++) {
        objNode = objNodeList.nextSibling();
        WScript.Echo(objNode.text);
    }
}
```

Output

Gambardella, Matthew

Ralls, Kim

Corets, Eva

...

C/C++ Syntax

 Copy

```
HRESULT nextNode(
    IXMLDOMNode **nextItem);
```

Parameters

nextItem [out, retval]

The next node in the collection, or Null if there is no next node.

Return Values

S_OK

The value returned if successful.

`S_FALSE`

The value returned if the `nextItem` parameter is Null.

Remarks

The iterator initially points before the first node in the list so that the first call to the `nextNode` method returns the first node in the list.

This method returns Null when the current node is the last node or there are no items in the list. When the current node is removed from the list, subsequent calls to `nextNode` return Null. The iterator must be reset by calling the `reset` method.

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNode](#)

[reset Method \(IXMLDOMNodeMap\)](#)

[IXMLDOMNodeMap](#)

[IXMLDOMNodeList](#)

[IXMLDOMSelection](#)

nextNode Method (IXMLDOMNodeNamedNodeMap)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Returns the next node in the collection.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeNamedNodeMap.nextNode();
```

Return Value

An `IXMLDOMNode`, which refers to the next node in the collection. Returns Null if there is no next node.

Example

The following script example creates an `IXMLDOMNodeNamedNodeMap` object and uses its `nextNode` method to iterate the collection.

Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var objNamedNodeMap;
var objNode;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    objNamedNodeMap = xmlDoc.documentElement.childNodes(1).attributes;
    for (var i=0; i<objNamedNodeMap.length; i++) {
        objNode = objNamedNodeMap.nextNode();
        WScript.Echo(objNode.text);
    }
}
```

Output

 Copy

```
bk102
```

C/C++ Syntax

 Copy

```
HRESULT nextNode(
    IXMLDOMNode **nextItem);
```

Parameters

`nextItem`[out, retval]

The next node in the collection, or Null if there is no next node.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value when returning Null.

E_INVALIDARG

The value returned if the `nextItem` parameter is Null.

Remarks

The iterator initially points before the first node in the list so that the first call to the `nextNode` method returns the first node in the list.

This method returns Null when the current node is the last node or there are no items in the list. When the current node is removed from the list, subsequent calls to `nextNode` return Null. The iterator must be reset by calling the `reset` method.

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNode](#)

[reset Method \(IXMLDOMNodeNamedNodeMap\)](#)

[IXMLDOMNodeNamedNodeMap](#)

[IXMLDOMNodeList](#)

[IXMLDOMSelection](#)

nodeFromID Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Returns the node that matches the ID attribute.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMDocument.nodeFromID(idString);
```

Parameters

`idString`

The string containing the value of the ID to match.

Return Value

An object. Returns the node that matches the supplied ID. If no nodes match, returns Null.

C/C++ Syntax

 Copy

```
HRESULT nodeFromID(  
    BSTR idString,
```

```
IXMLDOMNode **node);
```

Parameters

`idString[in]`

The value of the ID to match.

`node [out, retval]`

The node that matches the supplied ID. If no nodes match, this method returns Null.

Return Values

`S_OK`

The value returned if successful.

`S_FALSE`

The value returned when there is no node with the given ID.

`E_INVALIDARG`

The value returned if the `node` parameter is Null.

Remarks

According to the XML 1.0 Recommendation (REC-xml-19980210), ID attribute values must be unique within their XML documents and no element can specify more than one ID attribute.

The `nodeFromID` method was designed to handle ID and IDREF relationships in XML, but does not require an attribute of type IDREF. It can be used generically, and is similar to the `all` collection in DHTML.

To reference a node with `nodeFromID`, the node must be typed as ID in the schema or document type definition (DTD). Simply naming an attribute "ID" does not set its data type.

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMDocument-DOMDocument](#)

normalize Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Normalizes all descendant elements by combining two or more adjacent text nodes into one unified text node.

JScript Syntax

 Copy

```
oXMLDOMElement.normalize();
```

Example

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeRoot;
xmlDoc.async = false;
xmlDoc.loadXML("<root/>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeRoot = xmlDoc.documentElement;
    nodeRoot.appendChild(xmlDoc.createTextNode("Hello "));
    nodeRoot.appendChild(xmlDoc.createTextNode("World"));
    nodeRoot.appendChild(xmlDoc.createTextNode("!"));
    WScript.Echo(nodeRoot.childNodes.length);
    nodeRoot.normalize();
```

```
WScript.Echo(nodeRoot.childNodes.length);
}
```

Output

 Copy

```
3
1
```

C/C++ Syntax

 Copy

```
HRESULT normalize(void);
```

Return Values

S_OK

The value returned if successful.

E_FAIL

The value returned if an error occurs.

Example

```
c++

BOOL DOMElementNormalize()
{
    BOOL bResult = FALSE;
    IXMLElement *pIXMLElement = NULL;
    IXMLDOMDocument *pIXMLDOMDocument = NULL;
    HRESULT hr;

    try
    {
        // Create an instance of DOMDocument and initialize
        // pIXMLDOMDocument.
        // Load/create an XML fragment.
        hr = pIXMLDOMDocument->get_documentElement(&pIXMLElement);
        SUCCEEDED(hr) ? 0 : throw hr;
    }
}
```

```
if(pIXMLDOMELEMENT )
{
    hr = pIXMLDOMELEMENT->normalize();
    if(SUCCEEDED(m_hr))
        bResult = TRUE;
    pIXMLDOMELEMENT->Release();
}
catch(...)
{
    if(pIXMLDOMELEMENT)
        pIXMLDOMELEMENT->Release();
    DisplayErrorToUser();
}
return bResult;
}
```

Remarks

This method converts all text node descendants of this element (at any depth) into a "normal" form where no text nodes are adjacent. In normal form, text nodes can be separated only by markup, such as tags, comments, processing instructions, CDATA sections, and entity references. The normal form is useful for operations that require a particular document tree structure and ensures that the XML Document Object Model (DOM) view of a document is identical when saved and reloaded.

Collapsed node objects are deleted automatically.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0.

See Also

[IXMLDOMText](#)

[IXMLDOMELEMENT](#)

open Method (IXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Initializes an MSXML2.XMLHTTP request and specifies the method, URL, and authentication information for the request.

JScript Syntax

 Copy

```
oXMLHttpRequest.open(bstrMethod, bstrUrl, varAsync, bstrUser, bstrPassword);
```

Parameters

bstrMethod

The HTTP method used to open the connection, such as GET, POST, PUT, or PROPFIND. For XMLHTTP, this parameter is not case-sensitive. The verbs TRACE and TRACK are not allowed when IXMLHTTPRequest is hosted in the browser.

bstrUrl

The requested URL. This can be either an absolute URL, such as "http://Myserver/Mypath/Myfile.asp", or a relative URL, such as "../MyPath/MyFile.asp".

varAsync [optional]

A Boolean indicator of whether the call is asynchronous. The default is True (the call returns immediately). If set to True, attach an `onreadystatechange` property callback so that you can tell when the `send` call has completed.

bstrUser [optional]

The name of the user for authentication. If this parameter is Null ("") or missing and the site requires authentication, the component displays a logon window.

bstrPassword [optional]

The password for authentication. This parameter is ignored if the user parameter is Null ("") or missing.

Example

The following JScript example creates an `XMLHTTP` object, and then uses the `open` method to get a copy of `books.xml` at the IIS Web server running locally. The code then selects the `<book id="bk101">` element to display as output.

JavaScript

 Copy

```
var xmlhttp = new ActiveXObject("Msxml2.XMLHTTP.3.0");
xmlhttp.open("GET", "http://localhost/books.xml", false);
xmlhttp.send();
var book = xmlhttp.responseXML.selectSingleNode("//book[@id='bk101']");
WScript.Echo(book.xml);
```

C/C++ Syntax

 Copy

```
HRESULT open(BSTR bstrMethod, BSTR bstrUrl, VARIANT varAsync,
VARIANT bstrUser, VARIANT bstrPassword);
```

Parameters

bstrMethod [in]

The HTTP method used to open the connection, such as PUT or PROPFIND. For XMLHTTP, this parameter is not case-sensitive.

bstrUrl [in]

The requested URL. This can be either an absolute URL, such as "http://Myserver/Mypath/Myfile.asp", or a relative URL, such as "../MyPath/MyFile.asp".

`varAsync` [in, optional]

A Boolean indicator as to whether the call is asynchronous. The default is True (the call returns immediately). If set to True, attach an `onreadystatechange` property callback so that you can tell when the `send` call has completed.

`bstrUser` [in, optional]

The name of the user for authentication. If this parameter is Null ("") or missing and the site requires authentication, the component displays a logon window.

`bstrPassword` [in, optional]

The password for authentication. This parameter is ignored if the user parameter is Null or missing.

Return Values

`S_OK`

The value returned if successful.

Example

C++ Copy

```
HRESULT hr;
IXMLHttpRequest *pIXMLHttpRequest = NULL;

try
{
    // Create XMLHttpRequest object and initialize pIXMLHttpRequest.
    hr = pIXMLHttpRequest->open(_bstr_t(_T("PUT")),
        _bstr_t(_T("GET", "http://MyServer/Sample.xml", false)),
        _variant_t(VARIANT_FALSE), _variant_t(""), _variant_t(""));
    if(SUCCEEDED(hr))
        ::MessageBox(NULL, _T("Success !"), _T(""), MB_OK);
}
catch(...)
{
    DisplayErrorToUser();
}
// Release pIXMLHttpRequest when finished with it.
```

Remarks

After calling this method, you must call `send` to send the request and data, if any, to the server.

① Note

Although this method accepts credentials passed via parameter, those credentials are not automatically sent to the server on the first request. The `bstrUser` and `bstrPassword` parameters are not sent to the server unless the server challenges the client for credentials with a 401 - Access Denied response.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[abort Method \(IXMLHTTPRequest\)](#)

[onreadystatechange Property \(IXMLHTTPRequest\)](#)

[IXMLHTTPRequest](#)

open Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Initializes a request and specifies the method, URL, and authentication information for the request.

JScript Syntax

 Copy

```
oServerXMLHTTPRequest.open(bstrMethod, bstrUrl, bAsync, bstrUser,  
bstrPassword);
```

Parameters

bstrMethod

The HTTP method used to open the connection, such as PUT or PROPFIND. For ServerXMLHTTP, this parameter is case-sensitive and the method name must be entered in all upper-case letters.

bstrUrl

The requested URL. This can be either an absolute URL, such as "http://Myserver/Mypath/Myfile.asp", or a relative URL, such as "../MyPath/MyFile.asp".

bAsync (optional)

Boolean. Indicator as to whether the call is asynchronous. The default is False (the call does not return immediately).

bstrUser (optional)

The name of the user for authentication.

bstrPassword (optional)

The password for authentication. This parameter is ignored if the user parameter is Null or missing.

Example

The following JScript example creates an `XMLHTTP` object, and then uses the `open` method to get a copy of `books.xml` at the IIS Web server running locally. The code then selects the author and title information for the `<book id="bk101">` element to display as output.

JavaScript

 Copy

```
<%@language=JScript%>
<%
var srvXmlHttp
srvXmlHttp = Server.CreateObject("Msxml2.ServerXMLHTTP.3.0");
srvXmlHttp.open ("GET", "http://localhost/books.xml", false);
srvXmlHttp.send();
var catalog = srvXmlHttp.responseXML;
var author = catalog.selectSingleNode("//book[@id='bk101']/author");
var title = catalog.selectSingleNode("//book[@id='bk101']/title");
%>
<html>
<body>
<pre><code>
<%Response.Write("<p>BOOK SEARCH RESULTS:</p>");%
<%Response.Write("Author: " + author.text + "<br>");%
<%Response.Write("Title: " + title.text + "<br>");%
</code></pre>
</body>
</html>
```

C/C++ Syntax

 Copy

```
HRESULT open(BSTR bstrMethod, BSTR bstrUrl, VARIANT bAsync,  
VARIANT bstrUser, VARIANT bstrPassword);
```

Parameters

bstrMethod [in]

The HTTP method used to open the connection, such as PUT or PROPFIND. For ServerXMLHTTP, this parameter is case-sensitive and the method name must be entered in all upper-case letters.

bstrUrl [in]

The requested URL. This can be either an absolute URL, such as "http://Myserver/Mypath/Myfile.asp", or a relative URL, such as "../MyPath/MyFile.asp".

bAsync [in, optional]

Boolean. Indicator as to whether the call is asynchronous. The default is False (the call does not return immediately).

bstrUser [in, optional]

The name of the user for authentication. If this parameter is Null ("") or missing and the site requires authentication, the component displays a logon window.

bstrPassword [in, optional]

The password for authentication. This parameter is ignored if the user parameter is Null or missing.

Return Values

S_OK

The value returned if successful.

Remarks

After calling this method you must call the `send` method to send the request and data (if any) to the server. Each `send` method must have a corresponding `open` method.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[abort Method \(ServerXMLHTTP-I ServerXMLHTTPRequest\)](#)

[I ServerXMLHTTPRequest-ServerXMLHTTP](#)

Open Method (IXMLHttpRequest2, Windows 8)

10/27/2016 • 2 minutes to read

In this article

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Initializes an `IXMLHttpRequest2` request and specifies the method, URL, and authentication information for the request.

C/C++ Syntax

 Copy

```
HRESULT Open(const WCHAR *pwszMethod,  
const WCHAR *pwszUrl,  
IXMLHttpRequest2Callback *pStatusCallback,  
const WCHAR *pwszUserName,  
const WCHAR *pwszPassword,  
const WCHAR *pwszProxyUserName,  
const WCHAR *pwszProxyPassword  
);
```

Parameters

`pwszMethod` [in, string, ref]

The HTTP method used to open the connection, such as GET or POST.

`pwszUrl` [in, string, ref]

The requested URL. This can be either an absolute URL, such as "http://Myserver/Mypath/Myfile.asp".

`pStatusCallback` [in]

A callback function that is called when a callback event occurs.

`pwszUserName` [in, string, unique]

The name of the user for authentication. If this parameter is Null and the site requires authentication, the component displays a logon window unless the `SetProperty` method has been called to disable user input prompts, authentication, or both.

`pwszPassword` [in, string, unique]

The password for authentication. This parameter is ignored if the `pwszUserName` parameter is Null or missing.

`pwszProxyUserName` [in, string, unique]

The name of the user for authentication to the proxy server. If this parameter is Null and the proxy server requires authentication, the component displays a logon window unless the `SetProperty` method has been called to disable user input prompts, authentication, or both.

`pwszProxyPassword` [in, string, unique]

The password for authentication to the proxy server. This parameter is ignored if the `pwszProxyUserName` parameter is Null or missing.

Return Values

`S_OK`

The value returned if successful.

Example

```
c++ Copy
HRESULT hr;
IXMLHttpRequest *pIXMLHttpRequest = NULL;

try
{
    // Create XMLHttpRequest object and initialize pIXMLHttpRequest.
    hr = pIXMLHttpRequest->open(_bstr_t(_T("PUT")),
        _bstr_t(_T("GET"), "http://MyServer/Sample.xml", false)),
        _variant_t(VARIANT_FALSE), _variant_t(""), _variant_t(""));
    if(SUCCEEDED(hr))
        ::MessageBox(NULL, _T("Success !"), _T(""), MB_OK);
```

```
    }
    catch(...)
    {
        DisplayErrorToUser();
    }
// Release pIXMLHttpRequest when finished with it.
```

Remarks

After calling this method, you must call `Send` to send the request and data, if any, to the server.

ⓘ Note

Although this method accepts credentials passed via parameter, those credentials are not automatically sent to the server on the first request. The `pwszUserName` and `pwszPassword` parameters are not sent to the server unless the server challenges the client for credentials with a `401 - Access Denied` response.

Versioning

Implemented in: MSXML 6.0

See Also

[Abort Method \(IXMLHTTPRequest2, Windows 8\)](#)

[Send Method \(IXMLHTTPRequest2, Windows 8\)](#)

[SetProperty Method \(Windows 8, C++\) ↗](#)

[IXMLHTTPRequest2 \(Windows 8\)](#)

peekNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Gets the next node that the `nextNode` method will return without advancing the list position.

JScript Syntax

 Copy

```
var objXMLDOMNode = objXMLDOMSelection.peekNode();
```

Example

Note

You can use [books.xml](#) to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var oSelection, nodeBook;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
```

```
} else {
    oSelection = xmlDoc.selectNodes("//book");
    nodeBook = oSelection.peekNode();
    WScript.Echo(nodeBook.xml);
}
```

Output

 Copy

```
<book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications
    with XML.</description>
</book>
```

C/C++ Syntax

 Copy

```
HRESULT peekNode (IXMLDOMNode** ppNode);
```

Parameters

`ppNode [out, retval]`

The returned node, or Null if there are no more nodes or if E_PENDING is returned.

Return Values

`S_OK`

The value returned if the method is successful.

`E_PENDING`

The value returned if the context document is still being built and the selection object has hit the end of the available nodes to match.

Remarks

Like `nextNode`, `peekNode` does not result in a snapshot of all matching nodes. In MSXML 6.0, consecutive calls to `peekNode` will produce the same node over and over, independent of changes in the underlying tree. MSXML 3.0 does not return the same node if the underlying data is modified.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMSelection](#)

remove Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Visual Basic Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Removes the specified namespace from a collection.

JScript Syntax

 Copy

```
objXMLDOMSchemaCol.remove(namespaceURI);
```

Parameters

`namespaceURI`

The namespace to remove from the collection. This can be any string that can be used in an `xmlns` attribute, but it cannot contain entity references. The same white space normalization that occurs on the `xmlns` attribute occurs on this argument (that is, leading and trailing white space is trimmed, new lines are converted to spaces, and multiple adjacent white space characters are collapsed into one space).

Visual Basic Syntax

 Copy

```
objXMLDOMSchemaCol.remove  
(namespaceURI)
```

Parameters

namespaceURI

The namespace to remove from the collection. This can be any string that can be used in an `xmlns` attribute, but it cannot contain entity references. The same white space normalization that occurs on the `xmlns` attribute occurs on this argument (that is, leading and trailing white space is trimmed, new lines are converted to spaces, and multiple adjacent white space characters are collapsed into one space).

C/C++ Syntax

 Copy

```
HRESULT remove(BSTR namespaceURI);
```

Parameters

namespaceURI [in]

The namespace to remove from the collection.

This can be any string that can be used in an `xmlns` attribute, but it cannot contain entity references. The same white space normalization that occurs on the `xmlns` attribute occurs on this argument (that is, leading and trailing white space is trimmed, new lines are converted to spaces, and multiple adjacent white space characters are collapsed into one space).

Return Values

E_INVALIDARG

The value returned if the namespace does not exist in the collection.

E_FAIL

The value returned if the collection is read-only.

Remarks

In MSXML 6.0, when a schema is added to the SchemaCache, it can import schemas from other namespaces. Furthermore, all schemas added to the SchemaCache can be merged with schemas that have the same namespace. Therefore, the `remove` method no longer reverses the actions taken when you call the `add` method. The `remove` method is deprecated in MSXML 6.0, and throws a Not Implemented exception.

Versioning

Implemented in: MSXML 3.0, MSXML 4.0, and MSXML 5.0. Not implemented in MSXML 6.0.

See Also

[IXMLDOMSchemaCollection-XMLSchemaCache](#)

removeAll Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[Applies to](#)

Removes all the nodes from the collection described by the `IXMLDOMSelection`.

JScript Syntax

 Copy

```
objXMLDOMSelection.removeAll();
```

Example

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var oSelection;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.loadXML("<root><child/><child/><child/></root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    WScript.Echo("Before removing <child> nodes:\n" + xmlDoc.xml + "\n");
    oSelection = xmlDoc.selectNodes("//child");
    oSelection.removeAll();
    WScript.Echo("After removing <child> nodes:\n" + xmlDoc.xml + "\n");
}
```

This example outputs the books.xml file.

C/C++ Syntax

```
HRESULT removeAll();
```

 Copy

Return Values

S_OK

The value returned if the method is successful.

E_PENDING

The value returned if all nodes cannot be found at this time (in which case no nodes are removed).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

Applies to

[IXMLDOMSelection](#)

removeAttribute Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Removes or replaces the named attribute.

JScript Syntax

 Copy

```
oXMLElement.removeAttribute(name);
```

Parameters

`name`

A string specifying the name of the attribute to be removed or replaced.

Example

Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook;
```

```
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    WScript.Echo(nodeBook.attributes.length);
    nodeBook.removeAttribute("id");
    WScript.Echo(nodeBook.attributes.length);
}
```

Output

1
0

C/C++ Syntax

```
HRESULT removeAttribute(
    BSTR name);
```

Parameters

`name` [in]

The name of the attribute to be removed or replaced.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value returned when no attribute with the given name is found.

E_FAIL

The value returned if an error occurs.

Example

```
c++ Copy
IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;
_bstr_t bstrAttributeName = _T("dateCreated");
IXMLDOMDocument *pIXMLDOMDocument = NULL;
HRESULT hr;

try
{
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.
    // Load/create an XML fragment.
    hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
    SUCCEEDED(hr) ? 0 : throw hr;

    hr = pIXMLDOMELEMENT->removeAttribute(bstrAttributeName);
    if(SUCCEEDED(hr))
    {
        // Attribute removed.
    }
    pIXMLDOMELEMENT->Release();
    pIXMLDOMELEMENT = NULL;
    // Release pIXMLDOMDocument when finished with it.
}
catch(...)
{
    // Release pIXMLDOMDocument if it exists.
    if(pIXMLDOMELEMENT)
        pIXMLDOMELEMENT->Release();
    DisplayErrorToUser();
}
```

Remarks

If the specified attribute has a default value, this is equivalent to a replace operation: The current value is removed and a new attribute is created with the default value. This operation also resets the `specified` property of `IXMLDOMNode`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

specified Property

IXMLDOMElement

removeAttributeNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Removes the specified attribute from this element.

JScript Syntax

 Copy

```
var objXMLDOMAttribute = oXMLDOMELEMENT.removeAttribute(DOMAttribute);
```

Parameters

DOMAttribute

An object that supplies the `IXMLDOMAttribute` object to be removed from this element.

Return Value

An object. Returns the removed element.

Example

Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, nodeId;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodeId = nodeBook.getAttributeNode("id");
    WScript.Echo(nodeBook.attributes.length);
    nodeBook.removeAttributeNode(nodeId);
    WScript.Echo(nodeBook.attributes.length);
}
```

Output

1

0

C/C++ Syntax

 Copy

```
HRESULT removeAttributeNode(
    IXMLDOMAttribute *DOMAttribute,
    IXMLDOMAttribute **attributeNode);
```

Parameters

`DOMAttribute [in]`

The `DOMAttribute` object that is to be removed from this element.

`attributeNode [out, retval]`

The removed element.

Return Values

S_OK

The value returned if successful.

E_FAIL

The value returned if an error occurs.

Remarks

If the attribute has a default value, this call also creates a new attribute with the default value, associates the new attribute with this element, and resets the attribute's `specified` property.

Versioning

Implemented in: MSXML 3.0 and IMSXML 6.0

See Also

[specified Property](#)

[IXMLDOMAttribute](#)

[IXMLDOMElement](#)

removeChild Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Removes the specified child node from the list of children and returns it.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNode.removeChild(childNode);
```

Parameters

`childNodes`

An object. Child node to be removed from the list of children of this node.

Return Value

An object. Returns the removed child node.

Example

The following script example creates an `IXMLDOMNode` object (`currNode`), removes a child node from it, and displays the text of the removed node.

 Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var currNode;
var oldChild;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    currNode = root.childNodes.item(1);
    oldChild = currNode.removeChild(currNode.childNodes.item(1));
    WScript.Echo(oldChild.text);
}
```

Output

Midnight Rain

C/C++ Syntax

```
HRESULT removeChild(
    IXMLDOMNode *childNode,
    IXMLDOMNode **outOldChild);
```

 Copy

Parameters

`childNode` [in]

The child node to be removed from the list of children of this node.

`outOldChild` [out, retval]

The removed child node. If Null, the `childNode` object is not removed.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `oldChild` parameter is not a child of this node, when the specified `oldChild` is read-only and cannot be removed, or when `oldChild` is Null.

E_FAIL

The value returned if an error occurs.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMAttribute](#)

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMDocument-DOMDocument](#)

[IXMLDOMDocumentFragment](#)

[IXMLDOMDocumentType](#)

[IXMLDOMELEMENT](#)

[IXMLDOMEntity](#)

[IXMLDOMEntityReference](#)

[IXMLDOMNode](#)

[IXMLDOMNodeNotation](#)

[IXMLDOMProcessingInstruction](#)

[IXMLDOMText](#)

removeNamedItem Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Removes an attribute from the collection.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeMap.removeNamedItem(name);
```

Parameters

`name`

The string specifying the name of the attribute to remove from the collection.

Return Value

An object. Returns the node removed from the collection. Returns Null if the named node is not an attribute.

Example

Note

You can use [books.xml](#) to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    WScript.Echo(nodeBook.attributes.length);
    nodeBook.attributes.removeNamedItem("id");
    WScript.Echo(nodeBook.attributes.length);
}
```

Output

1

0

C/C++ Syntax

```
HRESULT removeNamedItem(
    BSTR name,
    IXMLDOMNode **namedItem);
```

Parameters

`name` [in]

The name of the attribute to remove from the collection.

`namedItem` [out, retval]

The node removed from the collection. Returns Null if the named node is not an attribute.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value when returning Null.

E_FAIL

The value returned if an error occurs.

Example

```
c++ Copy
IXMLDOMNode *pIXMLDOMNode = NULL;
IXMLDOMNamedNodeMap *pIXMLDOMNamedNodeMap = NULL;
BSTR bstrAttributeName = ::SysAllocString(_T("dateModified"));
IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;
VARIANT varValue;
HRESULT hr;
IXMLDOMDocument *pIXMLDOMDocument = NULL;

try
{
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.
    // Load/create an XML fragment.
    hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
    SUCCEEDED(hr) ? 0 : throw hr;

    if(pIXMLDOMELEMENT)
    {
        hr = pIXMLDOMELEMENT->get_attributes(&pIXMLDOMNamedNodeMap);
        if(SUCCEEDED(hr) && pIXMLDOMNamedNodeMap)
        {
            hr = pIXMLDOMNamedNodeMap->removeNamedItem(bstrAttributeName,
&pIXMLDOMNode);
            if(SUCCEEDED(hr) && pIXMLDOMNode)
            {
                pIXMLDOMNode->get_nodeValue(&varValue);
                ::MessageBox(NULL, _bstr_t(varValue), _T("Removed Item"), MB_OK);
                pIXMLDOMNode->Release();
                pIXMLDOMNode = NULL;
            }
            pIXMLDOMNamedNodeMap->Release();
            pIXMLDOMNamedNodeMap = NULL;
        }
        pIXMLDOMELEMENT->Release();
        pIXMLDOMELEMENT = NULL;
    }
    ::SysFreeString(bstrAttributeName);
    bstrAttributeName = NULL;
}
```

```
}

catch(...)

{
    if(bstrAttributeName)
        ::SysFreeString(bstrAttributeName);
    if(pIXMLDOMElement)
        pIXMLDOMElement->Release();
    if(pIXMLDOMNamedNodeMap)
        pIXMLDOMNamedNodeMap->Release();
    if(pIXMLDOMNode)
        pIXMLDOMNode->Release();
    DisplayErrorToUser();
}

// Release pIXMLDOMDocument when finished with it.
```

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNamedNodeMap](#)

removeNext Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Removes the next node.

JScript Syntax

 Copy

```
var objXMLDOMNode = objXMLDOMSelection.removeNext();
```

Example

Note

You can use [books.xml](#) to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
var oSelection;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    oSelection = xmlDoc.selectNodes("//book");
```

```
    while (oSelection.peekNode() != null) {
        oSelection.removeNext();
    }
    WScript.Echo(xmlDoc.xml);
}
```

Output

 Copy

```
<?xml version="1.0"?>
<catalog>
</catalog>
```

C/C++ Syntax

 Copy

```
HRESULT removeNext(IXMLDOMNode ** ppNode);
```

Parameters

`ppNode` [out, retval]

The node that was removed, or Null if there is no `nextNode` to remove. If the parameter is Null, the removed node is not returned, but is still removed.

Return Values

`S_OK`

The value returned if the method is successful.

`S_FALSE`

The value returned if no nodes left in the selection.

`E_PENDING`

The value returned if all nodes cannot be found at this time (in which case no nodes are removed).

Remarks

The `removeNext` method is equivalent to the following (except that it also works for attributes).

JavaScript

 Copy

```
var node = list.peekNode(); node.parentNode.removeChild(node);
```

The side effect is that the length of the collection is decremented and the `nextNode` and `item` methods will not return it because it has been removed.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMSelection](#)

removeQualifiedItem Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Removes the attribute with the specified namespace and attribute name.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNodeMap.removeQualifiedItem(baseName,  
namespaceURI);
```

Parameters

baseName

The string specifying the base name of the attribute, without namespace qualification.

namespaceURI

The string specifying the namespace prefix that qualifies the attribute name.

Return Value

An object. Returns the attribute node removed, or Null if no node was removed.

Example

ⓘ Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
var nodeBook;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    WScript.Echo(nodeBook.attributes.length);
    nodeBook.attributes.removeQualifiedItem("id", "");
    WScript.Echo(nodeBook.attributes.length);
}
```

Ouput

 Copy

```
1
0
```

C/C++ Syntax

 Copy

```
HRESULT removeQualifiedItem(
    BSTR baseName,
    BSTR namespaceURI,
    IXMLDOMNode **qualifiedItem);
```

Parameters

`baseName [in]`

The base name of the attribute, without namespace qualification.

`namespaceURI` [in]

The namespace prefix that qualifies the attribute name.

`qualifiedItem` [out, retval]

The attribute node removed, or Null if no node was removed.

Return Values

`S_OK`

The value returned if successful.

`S_FALSE`

The value when returning Null.

`E_FAIL`

The value returned if an error occurs.

Example

C++

 Copy

```
IXMLDOMNode *pIXMLDOMNode = NULL;
IXMLDOMNamedNodeMap *pIXMLDOMNamedNodeMap = NULL;
BSTR bstrAttributeName = ::SysAllocString(_T("dateModified"));
IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;
VARIANT varValue;
HRESULT hr;
IXMLDOMDocument *pIXMLDOMDocument = NULL;

try
{
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.
    // Load/create an XML fragment.
    hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
    SUCCEEDED(hr) ? 0 : throw hr;

    if(pIXMLDOMELEMENT)
    {
        hr = pIXMLDOMELEMENT->get_attributes(&pIXMLDOMNamedNodeMap);
        if(SUCCEEDED(hr) && pIXMLDOMNamedNodeMap)
        {
            hr = pIXMLDOMNamedNodeMap->removeQualifiedItem(bstrAttributeName,
NULL,
&pIXMLDOMNode);
            if(SUCCEEDED(hr) && pIXMLDOMNode)
            {
                pIXMLDOMNode->get_nodeValue(&varValue);
            }
        }
    }
}
```

```
    ::MessageBox(NULL, _bstr_t(varValue), _T("Removed Item"),
MB_OK);
    pIXMLDOMNode->Release();
    pIXMLDOMNode = NULL;
}
pIXMLDOMNamedNodeMap->Release();
pIXMLDOMNamedNodeMap = NULL;
}
pIXMLDOMELEMENT->Release();
pIXMLDOMELEMENT = NULL;
}
::SysFreeString(bstrAttributeName);
bstrAttributeName = NULL;
}
catch(...)
{
    if(bstrAttributeName)
        ::SysFreeString(bstrAttributeName);
    if(pIXMLDOMELEMENT)
        pIXMLDOMELEMENT->Release();
    if(pIXMLDOMNamedNodeMap)
        pIXMLDOMNamedNodeMap->Release();
    if(pIXMLDOMNode)
        pIXMLDOMNode->Release();
    DisplayErrorToUser();
}
// Release pIXMLDOMDocument when finished with it.
```

Remarks

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNamedNodeMap](#)

replaceChild Method

10/27/2016 • 3 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Replaces the specified old child node with the supplied new child node.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNode.replaceChild(newChild, oldChild);
```

Parameters

`newChild`

An object. The address of the new child that is to replace the old child. If Null, `oldChild` is removed without a replacement.

`oldChild`

An object. The address of the old child that is to be replaced by the new child.

Return Value

An object. Returns the old child that is replaced.

Example

The following example creates a new `IXMLDOMNode` object, `newElem`, and replaces the specified child node with `newElem`.

ⓘ Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var root;
var newElem;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    root = xmlDoc.documentElement;
    newElem = xmlDoc.createElement("PAGES");
    root.childNodes.item(1).replaceChild(newElem,
root.childNodes.item(1).childNodes.item(0));
    WScript.Echo(root.childNodes.item(1).xml);
}
```

Output

```
<book id="bk102">

<author>Ralls, Kim</author>

<title>Midnight Rain</title>

<genre>Fantasy</genre>

<price>5.95</price>

<publish_date>2000-12-16</publish_date>

<description>A former architect battles corporate zombies,
an evil sorceress, and her own childhood to become queen
of the world.</description>
```

C/C++ Syntax

 Copy

```
HRESULT replaceChild(  
    IXMLDOMNode *newChild,  
    IXMLDOMNode *oldChild,  
    IXMLDOMNode **outOldChild);
```

Parameters

`newChild` [in]

The address of the new child that is to replace the old child. If Null, the `oldChild` parameter is removed without a replacement.

`oldChild` [in]

The address of the old child that is to be replaced by the new child.

`outOldChild` [out, retval]

The old child that is replaced. If Null, no object is created.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `newChild` node cannot be inserted as a child of this node, when the specified `oldChild` is not a child of this node, or if the `oldChild` parameter is Null.

`E_FAIL`

The value returned if an error occurs.

Remarks

This operation depends on the value of the `nodeType` property.

NODE_ATTRIBUTE

This operation depends on the value of the `newChild` parameter:

<code>NODE_ATTRIBUTE</code> , <code>NODE_CDATA_SECTION</code> , <code>NODE_COMMENT</code> , <code>NODE_DOCUMENT</code> , <code>NODE_DOCUMENT_TYPE</code> , <code>NODE_ELEMENT</code> , <code>NODE_ENTITY</code> , <code>NODE_NOTATION</code> , <code>NODE_PROCESSING_INSTRUCTION</code>	Returns an error. These node types cannot be children of an attribute.
<code>NODE_DOCUMENT_FRAGMENT</code>	Replaces <code>oldChild</code> with the children of the document fragment in <code>newChild</code> and returns <code>oldChild</code> .
<code>NODE_ENTITY_REFERENCE</code> , <code>NODE_TEXT</code>	Replaces the specified <code>oldChild</code> with the supplied <code>newChild</code> and returns <code>oldChild</code> .
<code>NODE_CDATA_SECTION</code> , <code>NODE_COMMENT</code> , <code>NODE_ENTITY</code> , <code>NODE_NOTATION</code> , <code>NODE_PROCESSING_INSTRUCTION</code> , <code>NODE_TEXT</code>	Returns an error. These node types either cannot have children or their children are read-only.

NODE_DOCUMENT

This operation depends on the value of the `newChild` parameter.

<code>NODE_ATTRIBUTE</code> , <code>NODE_CDATA_SECTION</code> , <code>NODE_DOCUMENT</code> , <code>NODE_ENTITY</code> , <code>NODE_ENTITY_REFERENCE</code> , <code>NODE_NOTATION</code> , <code>NODE_TEXT</code>	Returns an error. These nodes are not valid as children of a document node.
<code>NODE_COMMENT</code> , <code>NODE_PROCESSING_INSTRUCTION</code>	Replaces the specified <code>oldChild</code> with the supplied <code>newChild</code> and returns <code>oldChild</code> .
<code>NODE_DOCUMENT_TYPE</code> , <code>NODE_ELEMENT</code>	Replaces <code>oldChild</code> with <code>newChild</code> and returns <code>oldChild</code> . By definition, an XML document (the document node) can have only a single child. Therefore, an error is returned if the document node already has a child.
<code>NODE_DOCUMENT_FRAGMENT</code>	Replaces the specified <code>oldChild</code> with the children of the document fragment (<code>newChild</code>) and returns <code>oldChild</code> . The insert operations are subject to the rules for child nodes and can fail if the document fragment children represent node types that cannot be inserted.

NODE_DOCUMENT_TYPE

Returns an error. The document type is read-only.

NODE_DOCUMENT_FRAGMENT

This operation depends on the value of the `newChild` parameter.

NODE_ATTRIBUTE,

NODE_DOCUMENT,

NODE_DOCUMENT_TYPE

Returns an error. These node types are not valid as children of a document fragment.

NODE_CDATA_SECTION,

NODE_COMMENT, NODE_ELEMENT,

NODE_ENTITY_REFERENCE,

NODE_PROCESSING_INSTRUCTION,

NODE_TEXT

Replaces the specified `oldChild` with the supplied `newChild` and returns `oldChild`.

NODE_DOCUMENT_FRAGMENT

Replaces the specified `oldChild` with the children of the document fragment (`newChild`) and returns `oldChild`. The insert operations are subject to the rules for child nodes and can fail if the document fragment children represent node types that cannot be inserted.

NODE_ENTITY, NODE_NOTATION

Returns an error. Entities and notations are read-only and cannot be inserted into a document.

NODE_ELEMENT

This operation depends on the value of the `newChild` parameter.

NODE_CDATA_SECTION,

NODE_COMMENT, NODE_ELEMENT,

NODE_ENTITY_REFERENCE,

NODE_TEXT,

NODE_PROCESSING_INSTRUCTION

Replaces the specified `oldChild` with `newChild` and returns `oldChild`.

NODE_ATTRIBUTE,

NODE_DOCUMENT,

NODE_DOCUMENT_TYPE,

NODE_ENTITY, NODE_NOTATION

Returns an error. These node types cannot be children of an element node.

NODE_DOCUMENT_FRAGMENT

Replaces the specified `oldChild` with the children of the document fragment (`newChild`) and returns `oldChild`. The insert operations are subject to the rules for child nodes and can fail if the document fragment children represent node types that cannot be inserted.

NODE_ENTITY_REFERENCE

Returns an error. Although the child nodes of an entity reference are the expanded entity, the children cannot be modified.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[nodeType Property1](#)
[IXMLDOMAttribute](#)
[IXMLDOMCDATASection](#)
[IXMLDOMCharacterData](#)
[IXMLDOMComment](#)
[IXMLDOMDocument-DOMDocument](#)
[IXMLDOMDocumentFragment](#)
[IXMLDOMDocumentType](#)
[IXMLDOMElement](#)
[IXMLDOMEntity](#)
[IXMLDOMEntityReference](#)
[IXMLDOMNode](#)
[IXMLDOMNodeText](#)
[IXMLDOMNotation](#)
[IXMLDOMProcessingInstruction](#)
[IXMLDOMText](#)

replaceData Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Replaces the specified number of characters with the supplied string.

JScript Syntax

 Copy

```
oXMLDOMCharacterData.replaceData(offset, count, data);
```

Parameters

`offset`

The long integer value specifying the offset, in characters, at which to start replacing string data.

`count`

The long integer value specifying the number of characters to replace.

`data`

The string containing the new data that replaces the old string data.

Example

The following script example creates a new `IXMLDOMComment` object, and then replaces the first five characters with a new string.

① Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var comment;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    comment = xmlDoc.createComment("Hello World!");
    WScript.Echo(comment.xml);
    comment.replaceData(0, 5, "Goodbye, Cruel");
    WScript.Echo(comment.xml);
}
```

Output

<!--Hello World!>

<!--Goodbye, Cruel World!>

C/C++ Syntax

 Copy

```
HRESULT replaceData(
    long offset,
    long count,
    BSTR data);
```

Parameters

`offset` [in]

The offset, in characters, at which to start replacing string data.

`count` [in]

The number of characters to replace.

`data[in]`

The new data that replaces the old string data.

Return Values

`S_OK`

The value returned if successful.

`S_FALSE`

The value when returning Null.

`E_FAIL`

The value returned if an error occurs.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMText](#)

reset Method (IXMLDOMNodeNamedNodeMap)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Resets the iterator.

JScript Syntax

 Copy

```
oXMLDOMNodeNamedNodeMap.reset();
```

Example

Note

You can use [books.xml](#) to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, nodeAttribute, s;
xmlDoc.setProperty("SelectionLanguage", "XPath");
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
```

```
WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodeBook.attributes.reset();
    nodeAttribute = nodeBook.attributes.nextSibling();
    WScript.Echo(nodeAttribute.text);
}
```

Output

bk101

C/C++ Syntax

 Copy

```
HRESULT reset(void);
```

Return Values

S_OK

The value returned if successful.

Remarks

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNodeNamedNodeMap](#)

reset Method (IXMLDOMNodeList)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Resets the iterator.

JScript Syntax

 Copy

```
oXMLDOMNodeList.reset();
```

Example

The following script example creates an `IXMLDOMNodeList` object and iterates the collection using the `nextNode` method. It then uses the `reset` method to reset the iterator to point before the first node in the list.

Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var objNodeList;
var objNode;
xmlDoc.async = false;
```

```
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    objNodeList = xmlDoc.getElementsByTagName("author");
    for (var i=0; i<objNodeList.length; i++) {
        objNode = objNodeList.nextNode();
        WScript.Echo(objNode.text);
    }
    objNodeList.reset();
    objNode = objNodeList.nextNode();
    WScript.Echo(objNode.text);
}
```

Output

 Copy

```
Gambardella, Matthew
Ralls, Kim
Corets, Eva
...
```

C/C++ Syntax

 Copy

```
HRESULT reset(void);
```

Return Values

S_OK

The value returned if successful.

Remarks

This method reinitializes the iterator to point before the first node in the `IXMLDOMNodeList` so that the next call to `nextNode` returns the first item in the list.

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[nextNode Method \(IXMLDOMNodeList\)](#)

[IXMLDOMNodeList](#)

[IXMLDOMSelection](#)

reset Method (IXMLDOMParseErrorCollection)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Example](#)

[Applies to](#)

[Versioning](#)

[See Also](#)

[This sample code uses features that were implemented only in MSXML 6.0.]

Resets the internal position to start, so that the `next` method will return the first error in the list.

JScript Syntax

 Copy

```
objIXMLDOMParseErrorCollection.reset();
```

C/C++ Syntax

 Copy

```
HRESULT reset  
( )
```

Parameters

None.

Return Values

S_OK

The value returned if successful. This method never fails.

Example

The following example illustrates the `reset` method. An attempt to validate XML data against an XML Schema results in two errors, because the XML file contains two invalid `<book>` elements. The code calls `next` on the resultant error collection object to walk through the collection, then calls the `reset` method and returns to the first error.

This example uses the same two resource files used in the allErrors example, books.xml and books.xsd. We've provided source files for the sample in three languages: JScript, Visual Basic, and C++. The output is the same in each language.

- [Resource Files \(books.xml and books.xsd\)](#)
- [JScript Code \(reset.js\)](#)
- [C/C++ Code \(reset.cpp\)](#)
- [Output for the reset Example](#)

Applies to

[IXMLDOMParseErrorCollection Interface](#)

Versioning

Implemented in: MSXML 6.0

See Also

[_newEnum Property](#)

[length Property \(IXMLDOMParseErrorCollection\)](#)

item Property

next Property1

reset Method (IXSLProcessor)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource File](#)

[Output](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Resets the state of the processor to the state it was in prior to calling the `transform` method.

JScript Syntax

 Copy

```
objXSLProcessor.reset();
```

Example

Note

You can use `books.xml` and `sample.xsl` (below) to run this sample code.

JavaScript

 Copy

```
var xslt = new ActiveXObject("Msxml2.XSLTemplate.6.0");
var xslDoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var xslProc;
xslDoc.async = false;
xslDoc.load("sample.xsl");
```

```

if (xslDoc.parseError.errorCode != 0) {
    var myErr = xslDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xslt.stylesheet = xslDoc;
    var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
    xmlDoc.async = false;
    xmlDoc.load("books.xml");
    if (xmlDoc.parseError.errorCode != 0) {
        var myErr = xmlDoc.parseError;
        WScript.Echo("You have error " + myErr.reason);
    } else {
        xslProc = xslt.createProcessor();
        xslProc.input = xmlDoc;
        xslProc.addParameter("param1", "Hello");
        xslProc.transform();
        WScript.Echo(xslProc.output);
        xslProc.reset();
        WScript.Echo(xslProc.output);
    }
}

```

Resource File

The JScript example uses the following file.

Sample.xsl

| | |
|---|------|
| XML | Copy |
| <pre> <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0"> <xsl:output method="html"/> <xsl:param name="param1"/> <xsl:template match="/"> The parameter value was: <xsl:value-of select="\$param1"/> </xsl:template> </xsl:stylesheet> </pre> | |

Output

The parameter value was: Hello

C/C++ Syntax

| | |
|--|------|
| | Copy |
|--|------|

```
HRESULT reset();
```

Remarks

It does not reset any other properties, such as `stylesheet` or `startMode`.

The `reset` method will cause any internally buffered output to be thrown away (that is, it aborts the asynchronous transformation). Calling `reset` results in the `readyState` value going from `READYSTATE_INTERACTIVE` back to `READYSTATE_LOADED`.

A re-entrant call to `reset` from inside the scope of a call to `transform` results in that `transform` call finishing as soon as it can, returning `E_FAIL` with the error message "User aborted transform."

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXSLProcessor](#)

save Method (DOMDocument)

10/27/2016 • 3 minutes to read

In this article

[Parameters](#)

[Example](#)

[Parameters](#)

[C/C++ Return Values](#)

[Example](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Saves an XML document to the specified location.

VB

 Copy

```
oXMLDOMDocument.save(destination);
```

Parameters

destination

An object. The object can represent a file name, an ASP `Response` object, a `DOMDocument` object, or a custom object that supports persistence. See [Remarks](#) for more information.

VB

 Copy

```
oXMLDOMDocument.save  
(destination)
```

Example

If your `DOMDocument` contains unformatted XML, you might want to format it before saving. There is no flag in `DOMDocument` that can be set to indent your XML. In this situation, you might want to use the SAX writer to produce the resulting XML.

c++

 Copy

```
HRESULT save(  
    VARIANT destination);
```

Parameters

destination[in]

The type of object to save. This object can represent a file name, an ASP `Response` object, an XML document object, or a custom object that supports persistence. See **Remarks** for more information.

C/C++ Return Values

S_OK

The value returned if successful.

XML_BAD_ENCODING

The value returned if the document contains a character that does not belong in the specified encoding. The character must use a numeric entity reference. For example, the Japanese Unicode character 20013 does not fit into the encoding Windows-1250 (the Central European alphabet) and therefore must be represented in markup as the numeric entity reference 中 or 中. This version of `save` does not automatically convert characters to the numeric entity references.

E_INVALIDARG

The value returned if a string was provided, but it is not a valid file name.

E_ACCESSDENIED

The value returned if a save operation is not permitted.

E_OUTOFMEMORY

The value returned if the save operation must allocate buffers.

(Other values)

Any other file system error can be returned in the `save(string)` case.

Example

C++

 Copy

```
BOOL DOMDocSaveLocation()
{
    BOOL bResult = FALSE;
    IXMLDOMDocument *pIXMLDOMDocument = NULL;
    HRESULT hr;

    try
    {
        _variant_t varString = _T("D:\\sample.xml");
        // Initialize pIXMLDOMDocument (create a DOMDocument).
        // Load document.
        hr = pIXMLDOMDocument->save(varString);
        if(SUCCEEDED(hr))
            bResult = TRUE;
    }
    catch(...)
    {
        DisplayErrorToUser();
        // Release the IXMLDOMDocument interface.
    }
    // Release the IXMLDOMDocument interface when finished with it.
    return bResult;
}
```

Remarks

The behavior differs based on the object specified by the `objTarget` parameter.

| Object | Description |
|--------|-------------|
|--------|-------------|

| Object | Description |
|--------|--|
| string | Specifies the file name. This must be a file name rather than a URL. The file is created, if necessary, and the contents are replaced entirely with the contents of the saved document. This mode is not intended for use from a secure client, such as Microsoft Internet Explorer. |

 Copy

```
dim xmldoc
  set xmldoc =
Server.CreateObject("Msxml2.DOMDocument.3.0")
  xmldoc.load("c:\myfile.xml")
  xmldoc.save(Server.MapPath("sample.xml"))
```

The ASP `Response` object sends the document back to the client that invoked the ASP script.

`IXMLDocument`
Object

Duplicates the original document. It is the equivalent of saving the document and reparsing it. The document goes through full persistence through XML markup, thereby testing the persistability of your XML document.

Custom
object
supporting
persistence

Any other custom COM object that supports `QueryInterface` for `IStream`, `IPersistStream`, or `IPersistStreamInit` can also be provided here, and the document will be saved accordingly. In the `IStream` case, the `IStream Write` method will be called as it saves the document; in the `IPersistStream` case, `IPersistStreamLoad` will be called with an `IStream` that supports the `Read`, `Seek`, and `Stat` methods.

External entity references in `<!DOCTYPE>`, `<!ENTITY>`, `<!NOTATION>`, and XML namespace declarations are not changed; they point to the original document. A saved XML document might not load if the URLs are not accessible from the location in which you saved the document.

Character encoding is based on the encoding attribute in the XML declaration, such as `<?xml version="1.0" encoding="windows-1252"?>`. When no encoding attribute is specified, the default setting is UTF-8.

Validation is not performed during `save`, which can result in an invalid document that does not load again because of a specified document type definition (DTD).

This member is an extension of the Worldwide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[Persistence and the DOM](#)

[IXMLDOMDocument-DOMDocument](#)

selectNodes Method

10/27/2016 • 3 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Applies the specified pattern-matching operation to this node's context and returns the list of matching nodes as `IXMLDOMNodeList`.

JScript Syntax

 Copy

```
var objXMLDOMNodeList = oXMLDOMNode.selectNodes(expression);
```

Parameters

expression

A string specifying an XPath expression.

Return Value

An object. Returns the collection of nodes selected by applying the given pattern-matching operation. If no nodes are selected, returns an empty collection.

Example

The following script example creates an `IXMLDOMNodeList` object containing the nodes specified by the *expression* parameter (for example, all the `<xsl:template>` nodes in an

XSLT style sheet). It then displays the number of nodes contained in the node list.

① Note

You can use hello.xsl in the [Hello World! \(XSLT\) topic](#) topic to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var objNodeList;
xmlDoc.async = false;
xmlDoc.load("hello.xsl");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xmlDoc.setProperty("SelectionNamespaces",
"xmlns:xsl='http://www.w3.org/1999/XSL/Transform'");
    xmlDoc.setProperty("SelectionLanguage", "XPath");
    objNodeList = xmlDoc.documentElement.selectNodes("//xsl:template");
    WScript.Echo(objNodeList.length);
}
```

Output

2

C/C++ Syntax

 Copy

```
HRESULT selectNodes(
    BSTR expression,
    IXMLDOMNodeList **resultList);
```

Parameters

expression[in]

A string specifying an XPath expression.

resultList[out, retval]

The list of nodes selected by applying the given pattern-matching operation. If no nodes

are selected, returns an empty node list.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `resultList` parameter is Null.

Example

The following Microsoft Visual C++ example creates an `IXMLDOMNodeList` object containing the nodes specified by the *expression* parameter (for example, all the `<xsl:template>` nodes in an XSLT stylesheet). It then displays the number of nodes contained in the node list.

XpathSelectNodes.cpp

```
c++ Copy
#include <msxml6.h>
using namespace MSXML2;

void dump_com_error(_com_error &e);

int main(int argc, char* argv[])
{
    CoInitialize(NULL);
    try{
        IXMLElementDocument2Ptr pXMLLoader = NULL;
        HRESULT hr = pXMLLoader.CreateInstance(__uuidof(DOMDocument60));

        // Set parser property settings
        pXMLLoader->async = VARIANT_FALSE;

        // Load the sample XML file
        hr = pXMLLoader->load("hello.xsl");

        // If document does not load report the parse error
        if(hr!=VARIANT_TRUE)
        {
            IXMLDOMParseErrorPtr pError;
            pError = pXMLLoader->parseError;
            _bstr_t parseError = _bstr_t("At line ") + _bstr_t(pError->Getline())
                + _bstr_t("\n") + _bstr_t(pError->Getreason());
        }
    }
}
```

```

    MessageBox(NULL,parseError, "Parse Error",MB_OK);
    return 0;
}
// Otherwise, build node list using SelectNodes
// and returns its length as console output
else
    pXMLDoc->setProperty("SelectionLanguage", "XPath");
    // Set the selection namespace URI if the nodes
    // you wish to select later use a namespace prefix
    pXMLDoc->setProperty("SelectionNamespaces",
    "xmlns:xsl='http://www.w3.org/1999/XSL/Transform'");
    IXMLDOMElementPtr pXMLElement = NULL;
    pXMLElement = pXMLDoc->documentElement;
    IXMLDOMNodeListPtr pXMLDomNodeList = NULL;
    pXMLDomNodeList = pXMLElement->selectNodes("//xsl:template");
    int count = 0;
    count = pXMLDomNodeList->length;
    printf("The number of <xsl:template> nodes is %i.\n", count);
}
catch(_com_error &e)
{
    dump_com_error(e);
}
return 0;
}

void dump_com_error(_com_error &e)
{
    printf("Error\n");
    printf("\a\tCode = %08lx\n", e.Error());
    printf("\a\tCode meaning = %s", e.ErrorMessage());
    _bstr_t bstrSource(e.Source());
    _bstr_t bstrDescription(e.Description());
    printf("\a\tSource = %s\n", (LPCSTR) bstrSource);
    printf("\a\tDescription = %s\n", (LPCSTR) bstrDescription);
}

```

Try It!

1. Set up a new empty project in Visual C++ for a Win32 console application.
For more information, see [Set Up My Visual C++ Project](#).
2. Copy the XpathSelectNodes.cpp file and paste it in as code for a new Visual C/C++ source file in your project.
3. Copy the hello.xsl file from the [Hello, World! \(XSLT\)](#) tutorial application and paste and save it as a resource file in the same folder as your project.

4. Build and execute the project as a console EXE application.

Results and Output

When run successfully using the sample code and files, the application will return as standard console output the following:

The number of <xsl:template> nodes is 2.

Remarks

For more information about using the `selectNodes` method with namespaces, see the [setProperty Method](#) topic.

The `selectSingleNode` method is similar to the `selectNodes` method, but returns only the first matching node rather than the list of all matching nodes.

`IXMLDOMNodeList` is live and immediately reflects changes to the nodes that appear in the list.

This member is an extension of the World Wide Web Consortium (W3C) Document Object Model (DOM).

ⓘ Note

Previously, in MSXML 3.0 and earlier versions, the selection object created by calling the `selectNodes` method would gradually calculate the node-set. If the DOM tree was modified, while the `selectNodes` call was still actively iterating its contents, the behavior could potentially change the nodes that were selected or returned. In MSXML 6., the node-set result is fully calculated at the time of selection. This ensures that the iteration is simple and predictable. In rare instances, this change might impact legacy code written to accommodate previous behavior.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

Using XSLT with the DOM or SAX

[selectSingleNode Method](#)

[IXMLDOMNodeList](#)

[IXMLDOMNodeAttribute](#)

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMDocument-DOMDocument](#)

[IXMLDOMDocumentFragment](#)

[IXMLDOMDocumentType](#)

[IXMLDOMElement](#)

[IXMLDOMEntity](#)

[IXMLDOMEntityReference](#)

[IXMLDOMNode](#)

[IXMLDOMNodeNotation](#)

[IXMLDOMProcessingInstruction](#)

[IXMLDOMText](#)

selectSingleNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Applies the specified pattern-matching operation to this node's context and returns the first matching node.

JScript Syntax

 Copy

```
var objXMLDOMNode = oXMLDOMNode.selectSingleNode(queryString);
```

Parameters

`queryString`

A string specifying an XPath expression.

Return Value

An object. Returns the first node that matches the given pattern-matching operation. If no nodes match the expression, returns a null value.

Example

The following script example creates an `IXMLDOMNode` object and sets it to the first instance of an AUTHOR node with a BOOK parent. It then displays the text of the node.

ⓘ Note

You can use `books.xml` to run this sample code.

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var currNode;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xmlDoc.setProperty("SelectionLanguage", "XPath");
    currNode = xmlDoc.selectSingleNode("//book/author");
    WScript.Echo(currNode.text);
}
```

Output

Gambardella, Matthew

C/C++ Syntax

 Copy

```
HRESULT selectSingleNode(
    BSTR queryString,
    IXMLDOMNode **resultNode);
```

Parameters

`queryString` [in]

A string specifying an XPath expression.

`resultNode` [out, retval]

The first node that is selected by the given pattern-matching operation. If no nodes match the expression, returns a null value.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value returned if there is no match.

E_INVALIDARG

The value returned if the `resultNode` parameter is Null.

Remarks

The `selectSingleNode` method is similar to the `selectNodes` method, but returns only the first matching node rather than the list of all matching nodes.

This member is an extension of the World Wide Web Consortium (W3C) Document Object Model (DOM).

Versioning

Implemented in: MSXML 3.0 and MSXML6.0

See Also

[Using XSLT with the DOM or SAX](#)

[selectNodes Method](#)

[setProperty Method1](#)

[IXMLDOMNodeAttribute](#)

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMDocument-DOMDocument](#)

[IXMLDOMDocumentFragment](#)

[IXMLDOMDocumentType](#)

[IXMLDOMElement](#)

[IXMLDOMEntity](#)

[IXMLDOMEntityReference](#)

[IXMLDOMNode](#)

[IXMLDOMNode](#)

[IXMLDOMProcessingInstruction](#)

[IXMLDOMText](#)

send Method (IXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Sends an HTTP request to the server and receives a response.

JScript Syntax

 Copy

```
oXMLHttpRequest.send(varBody);
```

Parameters

`varBody`

The body of the message being sent with the request.

Example

JavaScript

 Copy

```
xmlhttp = new ActiveXObject("Msxml2.XMLHTTP.6.0");
xmlhttp.open("GET", "http://localhost/sample.xml", false);
xmlhttp.send();
WScript.Echo(xmlhttp.responseXML.xml);
```

C/C++ Syntax

 Copy

```
HRESULT send(VARIANT varBody);
```

Parameters

`varBody` [in, optional]

The body of the message being sent with the request.

Return Values

`S_OK`

The value returned if successful.

Remarks

This method is synchronous or asynchronous, depending on the value of the `bAsync` parameter in the `open` method call. If `open` is called with `bAsync == False`, this call does not return until the entire response is received or the protocol stack times out. If `open` is called with `bAsync == True`, this call returns immediately.

This method takes one optional parameter, which is the `requestBody` to use. The acceptable VARIANT input types are BSTR, SAFEARRAY of UI1 (unsigned bytes), `IDispatch` to an XML Document Object Model (DOM) object, and `IStream *`. You can use only chunked encoding (for sending) when sending `IStream *` input types. The component automatically sets the Content-Length header for all but `IStream *` input types.

If the input type is a BSTR, the response is always encoded as UTF-8. The caller must set a Content-Type header with the appropriate content type and include a `charset` parameter.

If the input type is a SAFEARRAY of UI1, the response is sent as is without additional encoding. The caller must set a Content-Type header with the appropriate content type.

If the input type is an XML DOM object, the response is encoded according to the encoding attribute on the <? declaration in the document. If there is no XML declaration or encoding attribute, UTF-8 is assumed.

If the input type is an `IStream` *, the response is sent as is without additional encoding. The caller must set a Content-Type header with the appropriate content type.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[open Method \(IXMLHTTPRequest\)](#)

[IXMLHTTPRequest](#)

send Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Return Values](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Sends an HTTP request to the server and receives a response.

JScript Syntax

 Copy

```
oServerXMLHttpRequest.send(varBody);
```

Parameters

`varBody`

The body of the message being sent with the request.

Example

HTML

 Copy

```
<%@language=JScript%>
<%
var srvXmlHttp
srvXmlHttp = Server.CreateObject("Msxml2.ServerXMLHTTP.3.0");
srvXmlHttp.open ("GET", "http://myserver/myresponse.asp", false);
srvXmlHttp.send();
```

```
newsElement = srvXmlHttp.responseXML.selectSingleNode("/news/story1");
%>

<html>
<body>
<p>Top News Story</p>
<%Response.write(newsElement.text);%>
</body>
</html>
```

Return Values

S_OK

The value returned if successful.

Remarks

This method is synchronous or asynchronous, depending on the value of the `bAsync` parameter in the `open` method call. If `open` is called with `bAsync == False`, this call does not return until the entire response is received or the protocol stack times out. If `open` is called with `bAsync == True`, this call returns immediately.

This method takes one optional parameter, which is the `requestBody` to use. The acceptable VARIANT input types are BSTR, SAFEARRAY of UI1 (unsigned bytes), `IDispatch` to an XML Document Object Model (DOM) object, and `IStream *`. You can only use chunked encoding (for sending) when sending `IStream *`input types. The component automatically sets the Content-Length header for all but `IStream *`input types.

If the input type is a BSTR, the response is always encoded as UTF-8. The caller must set a Content-Type header with the appropriate content type and include a `charset` parameter.

If the input type is a SAFEARRAY of UI1, the response is sent as is without additional encoding. The caller must set a Content-Type header with the appropriate content type.

If the input type is an XMLElement object, the response is encoded according to the encoding attribute on the <? XML declaration in the document. If there is no XML declaration or encoding attribute, UTF-8 is assumed.

If the input type is an `IStream *`, the response is sent as is without additional encoding. The caller must set a Content-Type header with the appropriate content type.

ⓘ Important

You cannot call multiple `send` methods for a single `open` method. Instead, for each `send` method, you must call a corresponding `open` method. For example:

```
osrvXmlHttp.open()
```

```
osrvXmlHttp.send()
```

```
osrvXmlHttp.open()
```

```
osrvXmlHttp.send()
```

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[open Method \(ServerXMLHTTP-I ServerXMLHTTPRequest\)](#)

[I ServerXMLHTTPRequest-ServerXMLHTTP](#)

Send Method (IXMLHttpRequest2, Windows 8)

10/27/2016 • 2 minutes to read

In this article

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Sends an HTTP request to the server and receives a response.

C/C++ Syntax

 Copy

```
HRESULT Send(ISequentialStream *pBody,  
ULONGLONG cbBody);
```

Parameters

`pBody` [in, optional]

The body of the message being sent with the request. This stream is read in order to upload data for non-GET requests. For requests that do not require uploading, set this parameter to NULL.

`cbBody` [in]

The length of the message being sent with the request. For requests that do not require uploading, set this parameter to 0.

Return Values

S_OK

The value returned if successful.

Remarks

Because this method is asynchronous, it returns immediately, before any request processing. Your application must listen for callbacks defined by IXMLHTTPRequest2Callback to receive progress notifications for the HTTP request.

Versioning

Implemented in: MSXML 6.0

See Also

[open Method \(IXMLHTTPRequest\)](#)

[IXMLHTTPRequest](#)

setAttribute Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Sets the value of the named attribute.

JScript Syntax

 Copy

```
oXMLElement.setAttribute(name, value);
```

Parameters

`name`

The string specifying the name of the attribute. If the attribute with that name already exists, its value is changed. If the attribute with that name does not exist, it is created.

`value`

The variant that supplies the value for the named attribute.

Example

Note

You can use `books.xml` to run this sample code.

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodeBook.setAttribute("PublishDate", String(Date()))
    WScript.Echo(nodeBook.getAttribute("PublishDate"));
}
```

Output

Outputs the current date and time in the following format:

Thu Jun 12 14:12:38 2003

(Thursday, June 12, 2003, 2:12:38 pm)

C/C++ Syntax

```
HRESULT setAttribute(
    BSTR name,
    VARIANT value);
```

Parameters

`name` [in]

The name of the attribute. If an attribute with that name already exists, its value is changed. If an attribute with that name does not exist, it is created.

`value` [in]

The value for the named attribute.

Return Values

S_OK

The value returned if successful.

E_FAIL

The value returned if an error occurs.

Example

See the example in the `getAttribute` method.

Remarks

If an attribute with the supplied name already exists, this method changes its value to the supplied `value` parameter. The supplied string is not parsed, so any markup, such as syntax intended to be recognized as an entity reference, is treated as literal text and must be appropriately escaped by the implementation when it is written out.

To assign an attribute value that contains entity references, the user must create `IXMLDOMAttribute` plus any `IXMLDOMText` and `IXMLDOMEntityReference` objects, build the appropriate subtree, and call the `setAttributeNode` method.

Versioning

MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMAttribute](#)

[IXMLDOMText](#)

[IXMLDOMEntityReference](#)

[setAttributeNode Method](#)

[IXMLDOMElement](#)

setAttributeNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Sets or updates the supplied attribute node on this element.

JScript Syntax

 Copy

```
var objXMLDOMAttribute = oXMLDOMELEMENT.XMLDOMELEMENT(DOMAttribute);
```

Parameters

`DOMAttribute`

An object that contains the attribute node to be associated with this element.

Return Value

An object. Returns Null unless the new attribute replaces an existing attribute with the same name, in which case this method returns the previous, replaced attribute node.

Example

 Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, nodePublishDate;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodePublishDate = xmlDoc.createAttribute("PublishDate");
    nodePublishDate.value = String(Date());
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodeBook.setAttributeNode(nodePublishDate);
    WScript.Echo(nodeBook.getAttribute("PublishDate"));
}
```

Output

Outputs the current date and time in the following format:

Thu Jun 12 14:12:38 2003

(Thursday, June 12, 2003, 2:12:38 pm)

C/C++ Syntax

```
HRESULT setAttributeNode(
    IXMLDOMAttribute *DOMAttribute,
    IXMLDOMAttribute **attributeNode);
```

 Copy

Parameters

`DOMAttribute [in]`

An attribute node that is to be associated with this element.

`attributeNode [out, retval]`

Null unless the new attribute replaces an existing attribute with the same name, in which

case this method returns the previous, replaced attribute node.

Return Values

S_OK

The value returned if successful.

E_FAIL

The value returned if an error occurs.

Example

See the example in the `getAttributeNode` method.

Remarks

You cannot add an existing attribute to an element until you first remove it from its previous element. Also, you cannot add a namespace-qualified attribute when it uses the same prefix as another attribute with a different `namespaceURI`.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[namespaceURI Property \(IXMLDOMNode\)](#)

[IXMLDOMELEMENT](#)

setNamedItem Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Adds the supplied node to the collection.

JScript Syntax

 Copy

```
var objXMLDOMNode = oIXMLDOMNodeMap.setNamedItem(newItem);
```

Parameters

newItem

The object containing the attribute to be added to the collection.

Return Value

An object. Returns the attribute successfully added to the collection.

Example

Note

You can use [books.xml](#) to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeBook, nodePublishDate;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodePublishDate = xmlDoc.createAttribute("PublishDate");
    nodePublishDate.value = String(Date());
    nodeBook = xmlDoc.selectSingleNode("//book");
    nodeBook.attributes.setNamedItem(nodePublishDate);
    WScript.Echo(nodeBook.getAttribute("PublishDate"));
}
```

Output

Outputs the current date and time in the following format:

Thu Jun 12 14:12:38 2003

(Thursday, June 12, 2003, 2:12:38 pm)

C/C++ Syntax

```
HRESULT setNamedItem(
    IXMLDOMNode *newItem,
    IXMLDOMNode **nameItem);
```

 Copy

Parameters

`newItem` [in]

An attribute to be added to the collection.

`nameItem` [out, retval]

An attribute successfully added to the collection. If Null, no object is created.

Return Values

S_OK

The value returned if successful.

E_INVALIDARG

The value returned if the `newItem` parameter is Null.

E_FAIL

The value returned if an error occurs.

Example

```
c++ Copy  
  
HRESULT hr;  
IXMLDOMDocument *pIXMLDOMDocument = NULL;  
IXMLDOMNode *pIXMLDOMNode = NULL;  
IXMLDOMNodeMap *pIXMLDOMNodeMap = NULL;  
BSTR bstrAttributeName = ::SysAllocString(_T("dateModified"));  
IXMLDOMAttribute *pIXMLDOMAttribute = NULL;  
IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;  
  
try  
{  
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.  
    // Load/create an XML fragment.  
    hr = m_pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);  
    SUCCEEDED(hr) ? 0 : throw hr;  
  
    if(pIXMLDOMELEMENT)  
    {  
        hr = pIXMLDOMELEMENT->get_attributes(&pIXMLDOMNodeMap);  
        if(SUCCEEDED(hr) && pIXMLDOMNodeMap)  
        {  
            hr = m_pIXMLDOMDocument->createAttribute(bstrAttributeName,  
&pIXMLDOMAttribute);  
            if(SUCCEEDED(hr) && pIXMLDOMAttribute)  
            {  
                hr = pIXMLDOMAttribute->put_nodeValue(_variant_t(_T("year 2000")));  
                hr = pIXMLDOMNodeMap->setNamedItem(pIXMLDOMAttribute,  
&pIXMLDOMNode);  
                if(SUCCEEDED(hr) && pIXMLDOMNode)  
                {  
                    pIXMLDOMNode->Release();  
                    pIXMLDOMNode = NULL;  
                }  
                pIXMLDOMAttribute->Release();  
                pIXMLDOMAttribute = NULL;  
            }  
            pIXMLDOMNodeMap->Release();  
            pIXMLDOMNodeMap = NULL;  
        }  
    }  
}
```

```
    }
    pIXMLDOMELEMENT->Release();
    pIXMLDOMELEMENT = NULL;
}
::SysFreeString(bstrAttributeName);
bstrAttributeName = NULL;
}
catch(...)
{
    if(bstrAttributeName)
        ::SysFreeString(bstrAttributeName);
    if(pIXMLDOMELEMENT)
        pIXMLDOMELEMENT->Release();
    if(pIXMLDOMNamedNodeMap)
        pIXMLDOMNamedNodeMap->Release();
    if(pIXMLDOMAttribute)
        pIXMLDOMAttribute->Release();
    if(pIXMLDOMNode)
        pIXMLDOMNode->Release();
    DisplayErrorToUser();
}
// Release pIXMLDOMDocument when finished with it.
```

Remarks

If an attribute already exists with the name in `IXMLDOMNode`, the supplied replaces the existing attribute. The attribute name appears in its `IXMLDOMNode` property.

If the `newItem` node type is not `NODE_ATTRIBUTE`, `setNamedItem` returns an error. For example, it is not possible to modify entities or notations, which are read-only.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMNode](#)

[IXMLDOMNamedNodeMap](#)

setOption Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Sets one of the following options:

- `SXH_OPTION_URL_CODEPAGE`
- `SXH_OPTION_ESCAPE_PERCENT_IN_URL`
- `SXH_OPTION_IGNORE_SERVER_SSL_CERT_ERROR_FLAGS`
- `SXH_OPTION_SELECT_CLIENT_SSL_CERT`

For more information about these options, see Remarks.

JScript Syntax

 Copy

```
oServerXMLHttpRequest.setOption(option, value);
```

Parameters

option

The option whose value is to be set.

value

The value to which the specified option is to be set.

C/C++ Syntax

 Copy

```
HRESULT setOption(SERVERXMLHTTP_OPTION option, VARIANT value);
```

Parameters

`option` [in]

The option whose value is to be set.

`value` [in]

The value to which the specified option is to be set.

Return Values

`S_OK`

Value returned if successful.

Remarks

The following table lists the options available to the `setOption` method.

0

`SXH_OPTION_URL_CODEPAGE`: By default, `CP_UTF8` is the code page used to convert the Unicode URL string (specified in the `open` method) to a single-byte representation.

The `SXH_OPTION_URL_CODEPAGE` option allows the client to override this default with a different code page value. The client should specify an unsigned integer value for the code page.

1

`SXH_OPTION_ESCAPE_PERCENT_IN_URL`: By default, escaping unsafe ANSI characters in the URL (for example, " " -> "%20") does not escape the % character itself.

The `SXH_OPTION_ESCAPE_PERCENT_IN_URL` option allows the client to change this behavior. The client should specify a Boolean True/False value for this option.

2

```
shx.setOption(2, shx.getOption(2)  
- SXH_SERVER_CERT_IGNORE_CERT_DATE_INVALID)
```

To turn off a flag, you subtract it from the default value, which is the sum of all flags. For example, to catch an invalid date in a certificate, you turn off the SXH_SERVER_CERT_IGNORE_CERT_DATE_INVALID flag as follows:

All certificate errors.

The date in the certificate is invalid or has expired.

Mismatch between the visited hostname and the certificate name being used on the server.

Malformed certificate such as a certificate with no subject name.

Unknown certificate authority

3

```
shx.setOption(3, "MSXML")
```

ⓘ Note

Option -1, `SXH_OPTION_URL`, is a read-only return value, and can therefore not be set by the `setOption` method. For more information about this option, see [getOption Method](#).

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[open Method \(ServerXMLHTTP-I ServerXMLHTTPRequest\)](#)
[I ServerXMLHTTPRequest-ServerXMLHTTP](#)

setProperty Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

This method is used to set [second-level properties](#) on the DOM object.

JScript Syntax

 Copy

```
objXMLDOMDocument2.setProperty(name, value);
```

Parameters

`name`

The name of the property to be set. For a list of properties that can be set using this method, see [second-level properties](#).

`value`

The value of the specified property. For a list of property values that can be set using this method, see [second-level properties](#).

Example

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var selection;
```

```
xmlDoc.loadXML("<Customer><Name>Microsoft</Name></Customer>");  
xmlDoc.setProperty("SelectionLanguage", "XPath");  
selection = xmlDoc.selectNodes("Customer/Name");  
WScript.Echo(selection.expr + " -- " + selection.item(0).xml);
```

Output

 Copy

```
Customer/Name -- <Name>Microsoft</Name>
```

C/C++ Syntax

 Copy

```
HRESULT SetProperty (BSTR name, VARIANT value);
```

Parameters

`name` [in]

The name of the property to be set. For a list of properties that can be set using this method, see [second-level properties](#) .

`value` [in]

The value of the specified property. For a list of property values that can be set using this method, see [second-level properties](#) .

Return Values

`S_OK`

Value returned if successful.

`E_FAIL`

Value returned if name or value is invalid.

Remarks

Examples of the second-level properties include `SelectionLanguage`, `ValidateOnParse`, `ServerHTTPRequest`.

This method may not be applied to the first-level DOM properties that are exposed directly on a DOM object. Examples of the first-level properties include `validateOnParse`, `async`, `parseError`, etc.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[selectSingleNode Method](#)

[selectNodes Method](#)

[Second-Level DOM Properties](#)

[getProperty Method \(IXMLDOMDocument2\)](#)

[IXMLDOMDocument2](#)

setProxy Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Specifies proxy configuration.

JScript Syntax

 Copy

```
oSrvXMLHTTPRequest.setProxy(proxySetting, varProxyServer, varBypassList);
```

Parameters

`proxySetting`

The proxy configuration whose value is to be set. For a list of available settings and a description of how to use those settings, see [Remarks](#).

`varProxyServer`

The name of a proxy server or a list of proxy server names.

`varBypassList`

The list of locally known host names or IP addresses for which you want to permit bypass of the proxy server.

C/C++ Syntax

 Copy

```
HRESULT setProxy(  
    SXH_PROXY_SETTING proxySetting,  
    VARIANT varProxyServer,  
    VARIANT varBypassList);
```

Parameters

`proxySetting` [in]

The proxy configuration whose value is to be set. For a list of available settings and a description of how to use those settings, see **Remarks**.

`varProxyServer` [in, optional]

The name of a proxy server or a list of proxy server names.

`varBypassList` [in, optional]

The list of locally known host names or IP addresses for which you want to permit bypass of the proxy server.

Return Values

`S_OK`

Value returned if successful.

`E_FAIL`

Value returned if name or value is invalid.

Remarks

The following table lists the settings available for use with the `setProxy` method.

| Setting | Description |
|---------|-------------|
|---------|-------------|

| Setting | Description |
|---------|---|
| 0 | <p><code>SXH_PROXY_SET_PRECONFIG</code></p> <p><code><P>- or -</P></code></p> <p><code>SXH_PROXY_SET_DEFAULT</code></p> <p>Currently, the <code>SXH_PROXY_SET_DEFAULT</code> shares the same settings level as <code>SXH_PROXY_SET_PRECONFIG</code>. It is intended, however, to support auto-proxy configuration features that might be included with future released products.</p> <p>The <code>SXH_PROXY_SET_PRECONFIG</code> option can be used to specify that a previously established static proxy configuration should be used. When this option is used, the configuration is taken from the Windows registry. This is the case if you used the WinHTTP proxy configuration utility, proxycfg.exe, to set proxy settings at the client computer.</p> |
| 1 | <p><code>SXH_PROXY_SET_DIRECT</code></p> <p>The <code>SXH_PROXY_SET_DIRECT</code> option can be used to specify that all HTTP and HTTPS servers should be accessed directly.</p> |
| 2 | <p><code>SXH_PROXY_SET_PROXY</code></p> <p>The <code>SXH_PROXY_SET_PROXY</code> option can be used to specify one or more proxy servers, and an optional bypass list. If a proxy is not specified for a given protocol, and the server is not in the bypass list, the server cannot be accessed.</p> |

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[setProxyCredentials Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)
[IServerXMLHTTPRequest-ServerXMLHTTP Members](#)

setProxyCredentials Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Specify proxy authentication credentials.

JScript Syntax

 Copy

```
oSrvXMLHTTPRequest.setProxyCredentials(username, password);
```

Parameters

`username`

The name of the user to be authenticated.

`password`

The password for the user to be authenticated.

Example

The following example shows how to set proxy credentials.

```
jscript
```

 Copy

```
var xmlhttp = new ActiveXObject("MSXML2.ServerXMLHTTP.6.0");
var sURL = "http://www.adventure-works.com";

try {
    xmlhttp.setProxy("2", "adventure-works", "<local>");
    xmlhttp.open("GET", sURL , false);
    xmlhttp.setProxyCredentials("adventure-works2\\proxyuser", "password1");

    xmlhttp.send();
    WScript.Echo("No exception, and ServerXMLHTTP.status is " +
    xmlhttp.status);
}
catch (e) {
    WScript.Echo("Exception caught. There should be an exception if the
credentials are wrong.");
    WScript.Echo(e.number + " " + e.description);
    if (xmlhttp.status != 407) {
        WScript.Echo("It should have returned 407 (not authorized), but
returned = " + xmlhttp.status);
    } else {
        WScript.Echo("With wrong credentials, status returned 407 as
expected");
    }
}
```

C/C++ Syntax

 Copy

```
HRESULT setProxyCredentials(
    BSTR bstrUserName,
    BSTR bstrPassword);
```

Parameters

bstrUserName [in]

The name of the user to be authenticated.

bstrPassword [in]

The password for the user to be authenticated.

Return Values

S_OK

Value returned if successful.

E_FAIL

Value returned if name or value is invalid.

Versioning

Implemented in: MSXML 6.0 and later

See Also

[setProxy Method \(ServerXMLHTTP-I ServerXMLHTTPRequest\)](#)

[I ServerXMLHTTPRequest-ServerXMLHTTP Members](#)

setRequestHeader Method (IXMLHTTPRequest)

10/27/2016 • 3 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Specifies the name of an HTTP header.

JScript Syntax

 Copy

```
oXMLHttpRequest.setRequestHeader(bstrHeader, bstrValue);
```

Parameters

`bstrHeader`

A string. A header name to set; for example, `"depth"`. This parameter should not contain a colon and should be the actual text of the HTTP header.

`bstrValue`

A string. The value of the header; for example, `"infinity"`.

Note

You must call the `open` method before you call this method. Otherwise, an error will occur.

Example

The following VBScript example posts a `DOMDocument` to an Active Server Page (ASP) on a server and returns the result as a new XML document.

form.htm

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="VBScript">
Function onLoad()
    Dim mydata, pi
    Set mydata = CreateObject("Msxml2.DOMDocument.6.0")
    Set pi = mydata.createProcessingInstruction("xml", "version='1.0'
encoding='UTF-8'")
    mydata.insertBefore pi, mydata.firstChild
End function
Function sendInfo()
    Dim MyHttp
    'Do validation of input data before sending it.
    If(Not(customerName.value = ""))  then
        With MyData.documentElement
            .getElementsByName("Name").item(0).text = customerName.value
            .getElementsByName("Phone").item(0).text = customerPhoneNum.value
        End With
        Set MyHttp=CreateObject("Msxml2.XMLHTTP.6.0")
        MyHttp.open "POST", "http://localhost/httpreqserver.asp", False
        'Simulate message sent by a custom user agent.
        MyHttp.setRequestHeader "User-Agent", "MyCustomUser"
        MyData.async = False
        MyHttp.send mydata.XMLDocument
        Document.Write MyHttp.responseText
    Else
        Document.Write "Invalid data."
    End If
End function
</SCRIPT>
</HEAD>
<BODY LANGUAGE="JScript" ONLOAD="Return onLoad()">
<TABLE BORDER="2" ALIGN="center">
<TR><TD WIDTH="150" ALIGN="center">
    <LABEL>Name</LABEL>
    </TD><TD>
        <INPUT NAME="customerName" TYPE="EDIT"/>
    </TD></TR>
<TR><TD WIDTH="150" align="CENTER">
    <LABEL>Telephone number</LABEL>
    </TD><TD>
        <INPUT NAME="customerPhoneNum" type="EDIT"/>
    </TD></TR>
</TABLE>

```

 Copy

```

</TD></TR>
</TABLE>

<TABLE ALIGN="CENTER">
<TR><TD WIDTH="150" ALIGN="CENTER">
<INPUT TYPE="BUTTON" VALUE="Send Information" ALIGN="CENTER"
ONCLICK="sendInfo()"/>
</TD></TR>
</TABLE>

</BODY>
</HTML>
<XML id="MyData">
<MyStructure>
<Name/>
<Phone/>
</MyStructure>
</XML>

```

ASP File (httpreqserver.asp)

 Copy

```

<%@LANGUAGE="Jscript"%>
<%
    Response.Expires = -1000;
    // Load the posted XML document.
    var doc = Server.CreateObject("Msxml2.DOMDocument.6.0");
    doc.async=false;
    doc.load(Request);
    var result = Server.CreateObject("Msxml2.DOMDocument.6.0");
    // Now process the order and build the result document.
    var userAgent = Request.ServerVariables("HTTP_User-Agent");
    var OutputString="Data for "+
        doc.documentElement.childNodes.item(0).text +
        " (" + doc.documentElement.childNodes.item(1).text +
        ") added";
    Response.ContentType = "text/xml";
    if(userAgent == "MyCustomUser")
    {
        result.loadXML("<result>" + OutputString +" </result>");
        var pi = result.createProcessingInstruction("xml", "version='1.0'");
        result.insertBefore(pi, result.firstChild);
        result.save(Response);
    }
    else
    {
        Response.Write("<P><B>" + OutputString+" </B></P>");
    }
%>

```

Try It!

To run this sample, you need access to a computer running Internet Information Services (IIS) 5.0 or later.

1. Copy the HTML code provided above, and paste it into Notepad.
2. Save the file a form.htm to a valid Web virtual directory, such as C:\Inetpub\wwwroot, on your Web server computer.

! Note

If you are not running IIS locally on your computer, locate the following line:

! Note

```
MyHttp.open "POST", "http://localhost/httpreqserver.asp", False
```

! Note

Substitute the name of the remote computer for the characters "localhost", and save the file.

3. Copy the ASP code provided above, and paste it into Notepad.
4. Save the file as httpreqserver.asp, in the same Web virtual directory you used in Step 1.
5. Open your browser to the Web URL location where you saved the sample HTML file, such as http://localhost/form.htm.
6. Enter the name and phone number information, and click Send Information to add/submit the XML to the ASP request server page.

Output

When run, the sample Web application should return as output in the browser the name and phone number you entered as input. To verify that this information was generated using the XML <result> node string from httpreqserver.asp, you can do the following:

1. From within Internet Explorer, click View, and then click Source.
2. You can observe that the source matches the string that was generated by the ASP request server page in this format:

```
XML
```

 Copy

```
<?xml version="1.0"?>
<result>Data for [name input] ([phone number input] )added.</result>
```

C/C++ Syntax

```
HRESULT setRequestHeader(BSTR bstrHeader, BSTR bstrValue);
```

Parameters

`bstrHeader` [in]

A header name to set; for example, `"depth"`. This parameter should not contain a colon and should be the actual text of the HTTP header.

`bstrValue` [in]

The value of the header; for example, `"infinity"`.

Return Values

`S_OK`

The value returned if successful.

Remarks

If another header already exists with this name, it is concatenated to the previous header.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[getResponseHeader Method \(IXMLHTTPRequest\)](#)

[IXMLHTTPRequest](#)

setRequestHeader Method (ServerXMLHTTP- IServerXMLHTTPRequest)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Specifies the name of an HTTP header.

JScript Syntax

 Copy

```
oXMLServerHTTPRequest.setRequestHeader(bstrHeader, bstrValue);
```

Parameters

`bstrHeader`

The header name to set; for example, `"depth"`. This parameter should not contain a colon and should be the actual text of the HTTP header.

`bstrValue`

The value of the header; for example, `"infinity"`.

C/C++ Syntax

 Copy

```
HRESULT setRequestHeader(BSTR bstrHeader, BSTR bstrValue);
```

Parameters

bstrHeader [in]

The header name to set; for example, "depth". This parameter should not contain a colon and should be the actual text of the HTTP header.

bstrValue [in]

The value of the header; for example, "infinity".

Return Values

S_OK

The value returned if successful.

Remarks

If another header already exists with this name, it is replaced.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[getResponseHeader Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[IServerXMLHTTPRequest-ServerXMLHTTP](#)

setStartMode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource File](#)

[Output](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Performs subsets of larger XSL Transformations (XSLT) by selecting the XSLT mode with which to start. This minimizes the amount of XSLT processing.

The default value of the start mode is the empty string, "".

JScript Syntax

 Copy

```
objXSLProcessor.setStartMode(mode, namespaceURI);
```

Parameters

`mode`

The desired mode as a string. It must be the base name part of the qualified name.

`namespaceURI` (optional)

The full namespace URI to fully qualify the start mode name.

Example

JavaScript

 Copy

```
var xslt = new ActiveXObject("Msxml2.XSLTemplate.6.0");
var xslDoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var xslProc;
xslDoc.async = false;
xslDoc.load("sample2.xsl");
xsl.stylesheet = xslDoc;
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
xmlDoc.async = false;
xmlDoc.load("books.xml");
xslProc = xslt.createProcessor();
xslProc.input = xmlDoc;
xslProc.setStartMode("view");
xslProc.transform();
WScript.Echo(xslProc.output);
```

Resource File

The JScript example uses the following file.

Sample2.xsl

XML

 Copy

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html"/>
  <xsl:param name="param1"/>
  <xsl:template match="/">
    Hello
  </xsl:template>
  <xsl:template match="/" mode="edit">
    In Edit Mode
  </xsl:template>
  <xsl:template match="/" mode="view">
    In View Mode
  </xsl:template>
</xsl:stylesheet>
```

Output

In View Mode

C/C++ Syntax

 Copy

```
HRESULT setStartMode(BSTR mode, BSTR namespaceURI);
```

Parameters

`mode` [in]

The desired mode as a string. It must be the base name part of the qualified name.

`namespaceURI` [in, optional]

The full namespace URI to fully qualify the start mode name.

Return Values

`S_OK`

Success

`E_FAIL`

The value returned if the value of the `readyState` property is `READYSTATE_INTERACTIVE`.

`E_INVALIDARG`

The value returned if the mode base name contains a colon character or is an invalid name.

Remarks

Using `setStartMode` is essentially the same as an XSLT style sheet that starts with the following rule.

 Copy

```
<xsl:template match="/">
  <xsl:apply-templates select="*" mode="{mode}" />
</xsl:template>
```

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXSLProcessor](#)

setTimeouts Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Specifies timeout settings for resolving the domain name, establishing the connection to the server, sending the data, and receiving the response. The timeout parameters of the `setTimeouts` method are specified in milliseconds, so a value of 1000 would represent 1 second. A value of zero represents an infinite timeout. There are four separate timeout parameters: `resolveTimeout`, `connectTimeout`, `sendTimeout`, and `receiveTimeout`. When calling the `setTimeouts` method, all four values must be specified. The timeouts are applied at the Winsock layer.

JScript Syntax

 Copy

```
oServerXMLHttpRequest.setTimeouts(resolveTimeout, connectTimeout,  
    sendTimeout, receiveTimeout);
```

Parameters

`resolveTimeout`

A long integer. The value is applied to mapping host names (such as "www.microsoft.com") to IP addresses; the default value is infinite, meaning no timeout.

`connectTimeout`

A long integer. The value is applied to establishing a communication socket with the target server, with a default timeout value of 60 seconds.

sendTimeout

A long integer. The value applies to sending an individual packet of request data (if any) on the communication socket to the target server. A large request sent to a server will normally be broken up into multiple packets; the send timeout applies to sending each packet individually. The default value is 30 seconds.

receiveTimeout

A long integer. The value applies to receiving a packet of response data from the target server. Large responses will be broken up into multiple packets; the receive timeout applies to fetching each packet of data off the socket. The default value is 30 seconds.

Example

jscript

 Copy

```
var xmlServerHttp = new ActiveXObject("Msxml2.ServerXMLHTTP.6.0");
var lResolve = 5 * 1000;
var lConnect = 5 * 1000;
var lSend = 15 * 1000;
var lReceive = 15 * 1000;
xmlServerHttp.setTimeouts(lResolve, lConnect, lSend, lReceive);
xmlServerHttp.open("GET", "http://localhost/sample.xml", false);
xmlServerHttp.send();
```

C/C++ Syntax

 Copy

```
HRESULT setTimeouts (long resolveTimeout, long connectTimeout,
                     long sendTimeout, long receiveTimeout)
```

Parameters

resolveTimeout [in]

A long integer. The value is applied to mapping host names (such as "www.microsoft.com") to IP addresses; the default value is infinite, meaning no timeout.

connectTimeout [in]

A long integer. The value is applied to establishing a communication socket with the target server, with a default timeout value of 60 seconds.

`sendTimeout` [in]

A long integer. The value applies to sending an individual packet of request data (if any) on the communication socket to the target server. A large request sent to a server will normally be broken up into multiple packets; the send timeout applies to sending each packet individually. The default value is 30 seconds.

`receiveTimeout` [in]

A long integer. The value applies to receiving a packet of response data from the target server. Large responses will be broken up into multiple packets; the receive timeout applies to fetching each packet of data off the socket. The default value is 30 seconds.

Return Values

S_OK

Value returned if successful.

Remarks

The `setTimeouts` method should be called before the `open` method. None of the parameters is optional.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[open Method \(ServerXMLHTTP-I ServerXMLHTTPRequest\)](#)

[I ServerXMLHTTPRequest - ServerXMLHTTP](#)

splitText Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Visual Basic Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Splits this text node into two text nodes at the specified offset and inserts the new text node into the tree as a sibling that immediately follows this node.

JScript Syntax

 Copy

```
var objXMLDOMText = oXMLDOMText.splitText(offset);
```

Parameters

offset

A long integer. The number of characters at which to split this text node into two nodes, starting from zero.

Return Value

An object. Returns the new text node.

Example

jscript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var nodeRoot, nodeText, newNodeText;
xmlDoc.async = false;
xmlDoc.loadXML("<root>Hello World!</root>");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    nodeRoot = xmlDoc.documentElement;
    nodeText = nodeRoot.firstChild;
    WScript.Echo(nodeRoot.childNodes.length);
    newNodeText = nodeText.splitText(6);
    WScript.Echo(nodeRoot.childNodes.length);
}
```

Output

 Copy

```
1
2
```

Visual Basic Syntax

 Copy

```
Set objXMLDOMText = oXMLDOMText.splitText
(offset)
```

Parameters

`offset`

A long integer. The number of characters at which to split this text node into two nodes, starting from zero.

Return Value

An object. Returns the new text node.

C/C++ Syntax

```
HRESULT splitText(  
    long offset,  
    IXMLDOMText **rightHandTextNode);
```

 Copy

Parameters

`offset [in]`

The number of characters at which to split this text node into two nodes, starting from zero.

`rightHandTextNode [out, retval]`

The new text node.

Return Values

`S_OK`

The value returned if successful.

`S_FALSE`

The value when returning Null.

`E_FAIL`

The value returned if an error occurs.

Example

```
c++  
  
IXMLDOMText *pIXMLDOMTextTemp = NULL;  
IXMLDOMText *pIXMLDOMText = NULL;  
IXMLDOMNode *pIXMLDOMNodeTemp = NULL;  
IXMLDOMELEMENT *pIXMLDOMELEMENT = NULL;  
IXMLDOMNode *pIXMLDOMNode = NULL;  
DOMNodeType DOMNodeTyp;  
VARIANT varValue;  
HRESULT hr;
```

 Copy

`try`

```

{
    // Create an instance of DOMDocument and initialize pIXMLDOMDocument.
    // Load/create an XML fragment.
    hr = pIXMLDOMDocument->get_documentElement(&pIXMLDOMELEMENT);
    SUCCEEDED(hr) ? 0 : throw hr;

    if(pIXMLDOMELEMENT)
    {
        hr = pIXMLDOMELEMENT->QueryInterface(IID_IXMLDOMNode, (void
            **)&pIXMLDOMNode);
        SUCCEEDED(hr) ? 0 : throw hr;

        pIXMLDOMELEMENT->Release();
        pIXMLDOMELEMENT = NULL;

        while(pIXMLDOMNode)
        {
            hr = pIXMLDOMNode->get_nodeType(&DOMNodeTyp);
            if(SUCCEEDED( hr ) && (DOMNodeTyp == NODE_TEXT))
            {
                hr = pIXMLDOMNode->QueryInterface(IID_IXMLDOMText,
                    (void**)&pIXMLDOMText);
                SUCCEEDED(hr) ? 0 : throw hr;

                if(pIXMLDOMText)
                {
                    hr = pIXMLDOMText->splitText(2, &pIXMLDOMTextTemp);
                    if(SUCCEEDED(hr) && pIXMLDOMTextTemp)
                    {
                        hr = pIXMLDOMTextTemp->get_nodeValue(&varValue);
                        if(SUCCEEDED(hr))
                            ::MessageBox(NULL, _bstr_t(varValue), _T("Notation Public
ID"), MB_OK);
                        pIXMLDOMTextTemp->Release();
                        pIXMLDOMTextTemp = NULL;
                    }
                    pIXMLDOMText->Release();
                    pIXMLDOMText = NULL;
                }
                pIXMLDOMNode->Release();
                pIXMLDOMNode = NULL;
                break;
            }
            pIXMLDOMNode->getFirstChild(&pIXMLDOMNodeTemp);
            pIXMLDOMNode->Release();
            pIXMLDOMNode = pIXMLDOMNodeTemp;
            pIXMLDOMNodeTemp = NULL;
        }
    }
}

catch(...)
{
    if(pIXMLDOMNodeTemp)
        pIXMLDOMNodeTemp->Release();
}

```

```
if(pIXMLDOMNode)
    pIXMLDOMNode->Release();
if(pIXMLDOMTextTemp)
    pIXMLDOMTextTemp->Release();
if(pIXMLDOMText)
    pIXMLDOMText->Release();
if(pIXMLDOMElement)
    pIXMLDOMElement->Release();
DisplayErrorToUser();
}
// Release pIXMLDOMDocument when finished with it.
```

Remarks

If you specify a negative offset or an offset greater than the length of the first text node, it will return Invalid Argument error. If you specify an offset equal to the length of the first text node, it will return a NULL text node but not insert it into the DOM tree. If you specify an offset of zero and the first text node is not empty the first text node becomes empty and the right hand text node contains the previous contents of the node.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMCDATASection](#)

[IXMLDOMText](#)

substringData Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Retrieves a substring of the full string from the specified range.

JScript Syntax

 Copy

```
strValue = oXMLDOMCharacterData.substringData(offset, count);
```

Parameters

offset

A long integer value indicating the offset, in characters, from the beginning of the string.

An offset of zero indicates copying from the start of the data.

count

A long integer value indicating the number of characters to retrieve from the specified offset.

Return Value

A string. Returns the substring.

Example

The following script example creates an `IXMLDOMComment` object (comment), and then uses the `substringData` method to retrieve the first five characters of the object.

ⓘ Note

You can use `books.xml` to run this sample code.

JavaScript

 Copy

```
var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
var comment;
var MyStr;
xmlDoc.async = false;
xmlDoc.load("books.xml");
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    comment = xmlDoc.createComment("Hello World!");
    MyStr = comment.substringData(0, 5);
    WScript.Echo(MyStr);
}
```

Output

Hello

C/C++ Syntax

```
HRESULT substringData(
    long offset,
    long count,
    BSTR *data);
```

Parameters

`offset` [in]

The offset, in characters, from the beginning of the string. An offset of zero indicates copying from the start of the data.

`count` [in]

The number of characters to retrieve from the specified offset.

`data` [out, retval]

The substring to return.

Return Values

S_OK

The value returned if successful.

S_FALSE

The value when returning Null.

Remarks

If the `offset` and `count` parameters indicate a range beyond the end of the string, the returned substring continues only until the end of the string data.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMText](#)

transform Method1

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Resource File](#)

[Output](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Starts the transformation process or resumes a previously failed transformation.

JScript Syntax

 Copy

```
boolValue = objXSLProcessor.transform();
```

Example

JavaScript

 Copy

```
var xslt = new ActiveXObject("Msxml2.XSLTemplate.6.0");
var xslDoc = new ActiveXObject("Msxml2.FreeThreadedDOMDocument.6.0");
var xslProc;
xslDoc.async = false;
xslDoc.load("sample2.xsl");
if (xslDoc.parseError.errorCode != 0) {
    var myErr = xslDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xslt.stylesheet = xslDoc;
    var xmlDoc = new ActiveXObject("Msxml2.DOMDocument.6.0");
    xmlDoc.async = false;
    xmlDoc.load("books.xml");
```

```
if (xmlDoc.parseError.errorCode != 0) {
    var myErr = xmlDoc.parseError;
    WScript.Echo("You have error " + myErr.reason);
} else {
    xslProc = xslt.createProcessor();
    xslProc.input = xmlDoc;
    xslProc.transform();
    WScript.Echo(xslProc.output);
}
}
```

Resource File

The JScript example uses the following file.

Sample2.xsl

XML

Copy

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <xsl:output method="html"/>
    <xsl:param name="param1"/>
    <xsl:template match="/">
        Hello
    </xsl:template>
    <xsl:template match="/" mode="edit">
        In Edit Mode
    </xsl:template>
    <xsl:template match="/" mode="view">
        In View Mode
    </xsl:template>
</xsl:stylesheet>
```

Output

Copy

```
Hello
```

C/C++ Syntax

Copy

```
HRESULT transform (VARIANT_BOOL* pDone);
```

Parameters

pDone[out, retval]

The returned success message.

Return Values

E_FAIL

The value returned if the value of the `readyState` property is not `READYSTATE_LOADED` or `READYSTATE_COMPLETE`, or if a transformation error is encountered, for example, a script engine error.

VARIANT_FALSE

The value returned if the end of the input tree is reached and more input is pending.

VARIANT_TRUE

The value returned if the entire transformation is completed successfully.

Remarks

When the end of the transformation is reached successfully, `VARIANT_TRUE` is returned.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[IXSLProcessor](#)

transformNode Method

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[Visual Basic Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Processes this node and its children using the supplied XSL Transformations (XSLT) style sheet and returns the resulting transformation.

JScript Syntax

 Copy

```
strValue = oXMLDOMNode.transformNode(objStylesheet);
```

Parameters

objstylesheet

An object. A valid XML document or DOM node that consists of XSLT elements that direct the transformation of this node.

Return Value

A string. Returns a string that contains the product of the transformation of this XML document based on the XSLT style sheet.

Example

The following Microsoft JScript example shows the result of `transformNode`.

ⓘ Note

You can use `books.xml` and `sample2.xml` (below) to run this sample code.

JavaScript

 Copy

```
// Load data.  
var source = new ActiveXObject("Msxml2.DOMDocument.3.0");  
source.async = false;  
source.load("books.xml");  
if (source.parseError.errorCode != 0) {  
    var myErr = source.parseError;  
    WScript.Echo("You have error " + myErr.reason);  
} else {  
    // Load style sheet.  
    var stylesheet = new ActiveXObject("Msxml2.DOMDocument.3.0");  
    stylesheet.async = false  
    stylesheet.load("sample.xsl");  
    if (stylesheet.parseError.errorCode != 0) {  
        var myErr = stylesheet.parseError;  
        WScript.Echo("You have error " + myErr.reason);  
    } else {  
        // Echo back XSLT output to console  
        WScript.Echo(source.transformNode(stylesheet));  
    }  
}
```

Visual Basic Syntax

 Copy

```
strValue = oXMLDOMNode.transformNode  
(objStylesheet)
```

Parameters

`objstylesheet`

An object. A valid XML document or DOM node that consists of XSLT elements that direct the transformation of this node.

Return Value

A string. Returns a string that contains the product of the transformation of this XML document based on the XSLT style sheet.

Example

The following Microsoft Visual Basic example displays the result of `transformNode`.

VB

 Copy

```
Dim source As New Msxml2.DOMDocument30
Dim stylesheet As New Msxml2.DOMDocument30

' Load data.
source.async = False
source.Load App.Path & "\books.xml"
If (source.parseError.errorCode <> 0) Then
    Dim myErr
    Set myErr = source.parseError
    MsgBox("You have error " & myErr.reason)
Else
    ' Load style sheet.
    stylesheet.async = False
    stylesheet.Load App.Path & "\sample.xsl"
    If (stylesheet.parseError.errorCode <> 0) Then
        Dim myErr
        Set myErr = stylesheet.parseError
        MsgBox("You have error " & myErr.reason)
    Else
        ' Do the transform.
        MsgBox source.transformNode(stylesheet)
    End If
End If
```

C/C++ Syntax

 Copy

```
HRESULT transformNode(
    IXMLDOMNode *stylesheet,
    BSTR *xmlString);
```

Parameters

`stylesheet`[in]

A valid XML document or DOM node that consists of XSLT elements that direct the transformation of this node.

`xmlString`[out, retval]

A string that contains the product of the transformation of this XML document based on the XSLT style sheet.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `stylesheet` or `xmlString` parameter is Null.

Remarks

The `stylesheet` parameter must be either a `DOMDocument` node, in which case the document is assumed to be an Extensible Stylesheet Language (XSL) style sheet, or a Document Object Model (DOM) node in the XSL style sheet, in which case this node is treated as a standalone style sheet fragment.

The source node defines a context in which the style sheet operates, but navigation outside this scope is allowed. For example, a style sheet can use the `id` function to access other parts of the document.

This method supports both standalone and embedded style sheets and also provides the ability to run a localized style sheet fragment against a particular source node.

This member is an extension of the World Wide Web Consortium (W3C) DOM.

For more information about XSLT style sheets, see the [XSLT Reference](#).

Versioning

Implemented in: MSXML 3.0 and later

See Also

Using XSLT with the DOM or SAX

[IXMLDOMAttribute](#)

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMDocument-DOMDocument](#)

[IXMLDOMDocumentFragment](#)

[IXMLDOMDocumentType](#)

[IXMLDOMElement](#)

[IXMLDOMEntity](#)

[IXMLDOMEntityReference](#)

[IXMLDOMNode](#)

[IXMLDOMNodeotation](#)

[IXMLDOMProcessingInstruction](#)

[IXMLDOMText](#)

transformNodeToObject Method

10/27/2016 • 3 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

Remarks

Versioning

See Also

Processes this node and its children using the supplied XSL Transformations (XSLT) style sheet and returns the resulting transformation in the supplied object.

JScript Syntax

 Copy

```
oXMLDOMNode.transformNodeToObject(stylesheets, outputObject);
```

Parameters

`stylesheet`

An object. A valid XML document or DOM node that consists of XSLT elements that direct the transformation of this node.

`outputObject`

An object. On return, contains the product of the transformation of this XML document based on the XSLT style sheet. If the variant represents the `DOMDocument` object, the document is built according to its properties and its child nodes are replaced during this transformation process. The XML transformation can also be sent to a stream.

Example

The following Microsoft JScript example sets up the new XML document object named `result` before making the call to `transformNodeToObject`.

ⓘ Note

You can use `hello.xml` and `hello.xsl` in the [Hello World! \(XSLT\) topic](#) topic to run this sample code.

jscript

 Copy

```
// Load data.  
var source = new ActiveXObject("Msxml2.DOMDocument.6.0");  
source.async = false;  
source.load("hello.xml");  
if (source.parseError.errorCode != 0) {  
    var myErr = source.parseError;  
    WScript.Echo("You have error " + myErr.reason);  
} else {  
    // Load style sheet.  
    var stylesheet = new ActiveXObject("Msxml2.DOMDocument.6.0");  
    stylesheet.async = false;  
    stylesheet.load("hello.xsl");  
    if (stylesheet.parseError.errorCode != 0) {  
        var myErr = stylesheet.parseError;  
        WScript.Echo("You have error " + myErr.reason);  
    } else {  
        // Set up the resulting document.  
        var result = new ActiveXObject("Msxml2.DOMDocument.6.0");  
        result.async = false;  
        result.validateOnParse = true;  
        // Parse results into a result DOM Document.  
        WScript.Echo(source.transformNodeToObject(stylesheet, result));  
    }  
}
```

C/C++ Syntax

```
HRESULT transformNodeToObject(  
    IXMLDOMNode *stylesheet,  
    VARIANT outputObject);
```

Parameters

`stylesheet` [in]

A valid XML document or DOM node that consists of XSL elements that direct the transformation of this node.

`outputObject` [in]

An object that contains the product of the transformation of this XML document based on the XSLT style sheet. If the variant represents `DOMDocument`, the document is built according to its properties and its child nodes are replaced during this transformation process. If the variant contains an `IStream` interface, the XML transformation is sent to this stream.

Return Values

`S_OK`

The value returned if successful.

`E_INVALIDARG`

The value returned if the `stylesheet` or `outputObject` parameter is Null.

Example

This example sets up the new `DOMDocument` object and puts it into a VARIANT before making the call to `transformNodeTo0bject`.

```
c++ Copy
// p is the XML source that is to be transformed, pXSL is the style sheet.
// Create an empty DOM document for the result. (Error checking omitted
// for brevity.)
hr = CoCreateInstance(CLSID_DOMDocument60, NULL, CLSCTX_INPROC_SERVER,
    IID_IXMLDOMDocument, (void**)&pDoc);
hr = pDoc->QueryInterface(IID_IDispatch, (void **) &pDisp);
vObject.vt = VT_DISPATCH; // the new object
vObject.pdispVal = pDisp;
hr = p->transformNodeTo0bject(pXSL, vObject); // Transformation is
// present in pDoc.
```

Remarks

The `stylesheet` parameter must be either a `DOMDocument` node, in which case the document is assumed to be an XSLT style sheet, or a DOM node in the XSLT style sheet, in which case

this node is treated as a standalone style sheet fragment.

The source node defines a context in which the style sheet operates, but navigation outside this scope is allowed. For example, a style sheet can use the `id` function to access other parts of the document.

This method supports both standalone and embedded style sheets and also provides the ability to run a localized style sheet fragment against a particular source node.

The `transformNodeToObject` method always generates a Unicode byte-order mark, which means it cannot be used in conjunction with other Active Server Pages (ASP) `Response.Write` or `Response.BinaryWrite` calls.

For more information about XSLT, see the [XSLT Reference](#).

This member is an extension of the World Wide Web Consortium (W3C) DOM.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[Using XSLT with the DOM or SAX](#)

[IXMLDOMAttribute](#)

[IXMLDOMCDATASection](#)

[IXMLDOMCharacterData](#)

[IXMLDOMComment](#)

[IXMLDOMDocument-DOMDocument](#)

[IXMLDOMDocumentFragment](#)

[IXMLDOMDocumentType](#)

[IXMLDOMElement](#)

[IXMLDOMEntity](#)

[IXMLDOMEntityReference](#)

[IXMLDOMNode](#)

[IXMLDOMNodeNotation](#)

[IXMLDOMProcessingInstruction](#)

[IXMLDOMText](#)

validate Method1

10/27/2016 • 3 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Remarks](#)

[Versioning](#)

[See Also](#)

Performs run-time validation on the currently loaded document using the currently loaded document type definition (DTD), schema, or schema collection.

MSXML 6.0 doesn't support XDR schemas; the `validate` method validates only against DTDs and XSD schemas.

JScript Syntax

 Copy

```
objXMLDOMDocument2.validate();
```

Return Value

A `ParseError` object indicating exactly what error occurred, if any.

Example

The following Microsoft JScript example shows how to validate at runtime.

jscript

 Copy

```
var xmldoc = new ActiveXObject("Msxml2.DOMDocument.3.0");
xmldoc.async = false
// This will validate on load because validateOnParse is set to true
```

```

// by default.
xmlDoc.load("http://server/myData.xml");
// You make a change to an attribute:
xmlDoc.documentElement.setAttribute("a", "123");
// Now you want to verify your document is still valid:
var err = xmlDoc.validate();
if (err.errorCode == 0)
{
    WScript.Echo("Document is valid");
}
else
{
    WScript.Echo("Validation error:" + err.reason);
}

```

C/C++ Syntax

 Copy

```
HRESULT validate(IXMLDOMParseError ** errorObj);
```

Return Values

An `IErrorInfo` containing a formatted error message indicating what went wrong, and one of the following HRESULTs (also returned in the `ErrorInfo.Number` property).

| Return value | Hexadecimal value | Description |
|--------------|-------------------|---|
| E_PENDING | 0x8000000A | Document <code>readyState</code> is not 4, indicating the document is not completely loaded. |
| S_OK | 0 | The document is valid according to DTD or schemas. |
| S_FALSE | 1 | The document is invalid according to DTD or schemas. For error information, see the returned <code>IXMLDOMParseError</code> object. |

The following possible validation errors returned with S_FALSE are listed, together with the `errorCode` value.

| Message ID | Error Code (Hex Value) | Message |
|------------|------------------------|---------|
|------------|------------------------|---------|

| Message ID | Error Code
(Hex Value) | Message |
|--------------------------|-----------------------------------|--|
| XML_E_NODTD | 0xC00CE224 | The <code>validate</code> method failed because a DTD or schema was not specified in the document. |
| XML_E_NOTWF | 0xC00CE223 | The <code>validate</code> method failed because the document does not contain exactly one root node. |
| XML_ENTITY_UNDEFINED | 0xC00CE002 | Reference to undefined entity '%1'. |
| XML_INFINITE_ENTITY_LOOP | 0xC00CE003 | Entity '%1' contains an infinite entity reference loop. |
| XML_NDATA_INVALID_PE | 0xC00CE004 | Cannot use the NDATA keyword in a parameter entity declaration. |
| XML_REQUIRED_NDATA | 0xC00CE005 | Cannot use a general parsed entity '%1' as the value for attribute '%2'. |
| XML_NDATA_INVALID_REF | 0xC00CE006 | Cannot use unparsed entity '%1' in an entity reference. |
| XML_EXTENT_IN_ATTR | 0xC00CE007 | Cannot reference an external general parsed entity '%1' in an attribute value. |
| XML_ELEMENT_UNDECLARED | 0xC00CE00D | The element '%1' is used but not declared in the DTD or schema. |
| XML_ELEMENT_ID_NOT_FOUND | 0xC00CE00E | The attribute '%1' references the ID '%2', which is not defined in the document. |
| XML_EMPTY_NOT_ALLOWED | 0xC00CE011 | Element cannot be empty according to the DTD or schema. |
| XML_ELEMENT_NOT_COMPLETE | 0xC00CE012 | Element content is incomplete according to the DTD or schema. |

| Message ID | Error Code
(Hex Value) | Message |
|--------------------------------|-----------------------------------|--|
| XML_ROOT_NAME_MISMATCH | 0xC00CE013 | The name of the top-most element must match the name of the DOCTYPE declaration. |
| XML_INVALID_CONTENT | 0xC00CE014 | Element content is invalid according to the DTD or schema. |
| XML_ATTRIBUTE_NOT_DEFINED | 0xC00CE015 | The attribute '%1' on this element is not defined in the DTD or schema. |
| XML_ATTRIBUTE_FIXED | 0xC00CE016 | Attribute '%1' has a value that does not match the fixed value defined in the DTD or schema. |
| XML_ATTRIBUTE_VALUE | 0xC00CE017 | Attribute '%1' has an invalid value according to the DTD or schema. |
| XML_ILLEGAL_TEXT | 0xC00CE018 | Text is not allowed in this element according to the DTD or schema. |
| XML_MULTI_FIXED_VALUES | 0xC00CE019 | An attribute declaration cannot contain multiple fixed values: '%1'. |
| XML_ELEMENT_UNDEFINED | 0xC00CE01C | Reference to undeclared element: '%1'. |
| ML_XMLNS_FIXED | 0xC00CE01E | Attribute '%1' must be a #FIXED attribute. |
| XML_REQUIRED_ATTRIBUTE_MISSING | 0xC00CE020 | Required attribute '%1' is missing. |
| XML_DTD_EXPECTING | 0xC00CE026 | Expecting: %1. |

Remarks

This method only validates fully loaded documents (`readyState == 4`).

The `validate` method returns an `IXMLDOMParseError` that is independent of the value returned by the `parseError` property on a document. Only the `errorCode` and `reason` properties of the returned value are set.

Unlike the `load` method, `validate` will fail if there is no DTD or schema applied to the document element. Therefore, `validate` will not be able tell you whether the document is just well-formed.

The `validate` method does not parse new schemas, but can import a schema from a `SchemaCache` associated with the document through the `schemas` property. If there is no schema for a given namespace, the elements in that namespace will not be validated.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[readyState Property \(DOMDocument\)](#)

[IXMLDOMParseError](#)

[IXMLDOMDocument2](#)

validate Method (IXMLDOMSchemaCollection2- XMLDOMSchemaCollection)

10/27/2016 • 2 minutes to read

In this article

[JScript Syntax](#)

[C/C++ Syntax](#)

[Versioning](#)

[See Also](#)

Performs run-time validation on the documents in the schema cache that have not been compiled and validated. If a schema has its `validateOnLoad` property set to `true` when it is loaded into the cache, the schema will be compiled and validated at that time. The `validate` method is used when `validateOnLoad` is set to `false`. Steps 2 and 3 are postponed until the `validate` method is called.

The schema compilation steps are:

1. Check Syntax and resolve names.
2. Load and preprocess external schemas.
3. Compile schema items respecting W3Cs rules.

MSXML 6.0 doesn't support XDR schemas; this method validates only against DTDs and XSD schemas.

JScript Syntax

 Copy

```
objXMLSchemaCache.validate();
```

Return Value

None.

C/C++ Syntax

 Copy

```
HRESULT validate();
```

Return Value

None.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[validateOnLoad Property](#)

[IXMLDOMDocument2](#)

validateNode Method

10/27/2016 • 2 minutes to read

In this article

- [JScript Syntax](#)
- [Visual Basic Syntax](#)
- [C/C++ Syntax Using Smart Pointers](#)
- [Remarks](#)
- [Example](#)
- [Applies to](#)
- [Versioning](#)
- [See Also](#)

[This sample code uses features that were first implemented in MSXML 5.0 for Microsoft Office Applications.]

Validates a specified DOM fragment. A DOM fragment is a sub-tree spanning from a given node.

JScript Syntax

 Copy

```
var objErr = objXMLDOMDocument3.validateNode(node);
```

Visual Basic Syntax

 Copy

```
Set objErr = objXMLDOMDocument3.validateNode(node);
```

C/C++ Syntax Using Smart Pointers

 Copy

```
IXMLDOMParseErrorPtr objErr = objXMLDOMDocument3->validateNode(node);
HRESULT IXMLDOMDocument3::validateNode(
...[in] IXMLDOMNode* node,
...[out, retval] IXMLDOMParseError** objErr);
```

Parameters

`node`

An `IXMLDOMNode` object representing the node from which the fragment or sub-tree expands.

`objErr`

A reference to the `IXMLDOMParseError` object that indicates exactly what error occurred, if any. Use the properties on this object to examine specific parse error information.

Return Values

`S_OK`

Validation was successful.

`E_FAIL`

The node failed to validate for one of the following reasons:

- The DOM document was not completely loaded
- The DOM node does not belong to this DOM document instance
- The DOM node type was not element or attribute.

`S_FALSE`

Validation errors were encountered. Check the returned `objErr` object for detailed parse error information.

Note

This method returns the same `HRESULT` value that is returned by the `IXMLDOMDocument2::import` method.

Remarks

In the syntax for Visual Basic and JScript above, an `objXMLDOMDocument3` object is a `DOMDocument` instance that implements the `IXMLDOMDocument3` interface.

Unlike `IXMLDOMDocument2::validate`, `validateNode` will not cause a new schema to load. Furthermore, `validateNode` assumes that the rest of the document is valid. Without this assumption it would not be possible to validate ID/IDREFs at all with the `validateNode` method.

The `validateNode` method is applicable only to element or attribute nodes located within the same DOM document. Any attempt to call it on other types of nodes or nodes in other DOM documents results in an error.

Example

The `validateNode` example shows how to validate a DOM fragment at run time. The example uses two resource files, `validateNode.xml` and `validateNode.xsd`. The first is an XML data file, and the second is the XML Schema for the XML data. The first `<book>` element in `validateNode.xml` is valid against `validateNode.xsd`; the second `<book>` element is not.

We've provided source files for the sample in three languages: JScript, Visual Basic, and C++. The output is the same in each language.

- [Resource Files \(`validateNode.xml` and `validateNode.xsd`\)](#)
- [JScript Code \(`validateNode.js`\)](#)
- [Visual Basic Code \(`validateNode.frm`\) ↗](#)
- [C/C++ Code \(`validateNode.cpp`\)](#)
- [Output for `validateNode` Example](#)

Applies to

[IXMLDOMDocument3](#)

Versioning

Implemented in: MSXML 5.0 for Microsoft Office Applications and later

See Also

[IXMLDOMNode](#)

[IXMLDOMNodeList](#)

[IXMLDOMParseError](#)

[IXMLDOMSchemaCollection-XMLSchemaCache](#)

waitForResponse Method

10/27/2016 • 2 minutes to read

In this article

[Parameters](#)

[Example](#)

[Parameters](#)

[Parameters](#)

[C/C++ Return Values](#)

[Versioning](#)

[See Also](#)

Allows the requesting server to suspend execution while waiting for an asynchronous send operation to complete.

jscript

 Copy

```
oServerXMLHttpRequest.waitForResponse(timeoutInSeconds);
```

Parameters

timeoutInSeconds(optional)

Specifies the number of seconds to wait for an asynchronous send operation to complete.

Example

JavaScript

 Copy

```
var xmlServerHttp = new ActiveXObject("Msxml2.ServerXMLHTTP.6.0");
xmlServerHttp.open("GET", "http://localhost/sample.xml", true);
xmlServerHttp.send();
while (xmlServerHttp.readyState != 4) {
    xmlServerHttp.waitForResponse(1000);
}
```

Parameters

timeoutInSeconds(optional)

Specifies the number of seconds to wait for an asynchronous send operation to complete.

Parameters

timeoutInSeconds [in, optional]

Specifies the number of seconds to wait for an asynchronous send operation to complete.

isSuccessful [out, retval]

The `waitForResponse` method is more efficient than polling the `readyState` property, which is the only way to wait for an asynchronous send using the `XMLHTTP` component. The caller can supply an optional `timeout` parameter, specified in seconds. The default timeout (if one is not specified) is INFINITE (-1). The method returns True if a response is received within the time allotted, or False if a timeout occurs. If the method times out, the request is not aborted; the caller can continue to wait for the request in a subsequent call to the `waitForResponse` method. Calling `waitForResponse` after a synchronous `send` method returns immediately and has no useful effect. The `async` parameter to the `open` method controls whether the send is synchronous (default) or asynchronous. The results of this method are valid only after the `send` method has been completed successfully.

C/C++ Return Values

S_OK

The value returned if successful.

Versioning

Implemented in: MSXML 3.0 and MSXML 6.0

See Also

[open Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[readyState Property \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[send Method \(ServerXMLHTTP-IServerXMLHTTPRequest\)](#)

[IServerXMLHTTPRequest-ServerXMLHTTP](#)

