

[MS-DOM4]:

Microsoft Edge / Internet Explorer DOM4 Standards Support Document

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
3/22/2016	1.0	New	Released new document.
7/19/2016	1.1	Minor	Clarified the meaning of the technical content.
11/2/2016	1.1	None	No changes to the meaning, language, or formatting of the technical content.
3/14/2017	1.1	None	No changes to the meaning, language, or formatting of the technical content.
10/3/2017	1.1	None	No changes to the meaning, language, or formatting of the technical content.
2/22/2018	1.1	None	No changes to the meaning, language, or formatting of the technical content.
3/23/2018	1.1	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1	Introduction	4
1.1	Glossary	4
1.2	References	4
1.2.1	Normative References	4
1.2.2	Informative References	4
1.3	Microsoft Implementations	4
1.4	Standards Support Requirements	5
1.5	Notation	5
2	Standards Support Statements	6
2.1	Normative Variations	6
2.1.1	[W3C-DOM4] Section 3.2 Interface Event	6
2.1.2	[W3C-DOM4] Section 3.6 Interface EventTarget	8
2.1.3	[W3C-DOM4] Section 4.2.2 Interface NonElementParentNode	8
2.1.4	[W3C-DOM4] Section 4.2.3 Interface ParentNode	9
2.1.5	[W3C-DOM4] Section 4.2.4 Interface NonDocumentTypeChildNode	10
2.1.6	[W3C-DOM4] Section 4.5.1 Interface DOMImplementation	11
2.1.7	[W3C-DOM4] Section 4.8 Interface Element	12
2.1.8	[W3C-DOM4] Section 4.8.1 Interface Attr	12
2.1.9	[W3C-DOM4] Section 4.9 Interface CharacterData	13
2.1.10	[W3C-DOM4] Section 4.10 Interface Text	13
2.1.11	[W3C-DOM4] Section 5.2 Interface Range	14
2.1.12	[W3C-DOM4] Section 6.1 Interface NodeIterator	14
2.2	Clarifications	14
2.3	Error Handling	14
2.4	Security	15
3	Change Tracking	16
4	Index	17

1 Introduction

This document describes the level of support provided by Microsoft web browsers for the W3C *DOM4* specification [W3C-DOM4], published 19 November 2015. The [W3C-DOM4] specification defines a platform-neutral model for events and node trees.

1.1 Glossary

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the [Errata](#).

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <http://www.rfc-editor.org/rfc/rfc2119.txt>

[W3C-DOM4] World Wide Web Consortium, "W3C DOM4", W3C Recommendation 19 November 2015, <https://www.w3.org/TR/2015/REC-dom-20151119/>

1.2.2 Informative References

None.

1.3 Microsoft Implementations

The following Microsoft web browser versions implement some portion of the [W3C-DOM4] specification:

- Microsoft Edge

Each browser version may implement multiple document rendering modes. The modes vary from one to another in support of the standard. The following table lists the document modes supported by each browser version.

Browser Version	Document Modes Supported
Microsoft Edge	EdgeHTML Mode

For each variation presented in this document there is a list of the document modes and browser versions that exhibit the behavior described by the variation. All combinations of modes and versions that are not listed conform to the specification. For example, the following list for a variation indicates that the variation exists in three document modes in all browser versions that support these modes:

Quirks Mode, IE7 Mode, and IE8 Mode (All Versions)

1.4 Standards Support Requirements

To conform to [\[W3C-DOM4\]](#), a user agent must implement all required portions of the specification. Any optional portions that have been implemented must also be implemented as described by the specification. Normative language is usually used to define both required and optional portions. (For more information, see [\[RFC2119\]](#).)

The following table lists the sections of [W3C-DOM4] and whether they are considered normative or informative.

Sections	Normative/Informative
1-2	Informative
3-7	Normative
8	Informative
A, B	Informative

1.5 Notation

The following notations are used in this document to differentiate between notes of clarification, variation from the specification, and points of extensibility.

Notation	Explanation
C####	This identifies a clarification of ambiguity in the target specification. This includes imprecise statements, omitted information, discrepancies, and errata. This does not include data formatting clarifications.
V####	This identifies an intended point of variability in the target specification such as the use of MAY, SHOULD, or RECOMMENDED. (See [RFC2119] .) This does not include extensibility points.
E####	Because the use of extensibility points (such as optional implementation-specific data) can impair interoperability, this profile identifies such points in the target specification.

For document mode and browser version notation, see also section [1.3](#).

2 Standards Support Statements

This section contains all variations and clarifications for the Microsoft implementation of [\[W3C-DOM4\]](#).

- Section [2.1](#) describes normative variations from the MUST requirement of the specification.
- Section [2.2](#) describes clarifications of the MAY and SHOULD requirements.
- Section [2.3](#) considers error handling aspects of the implementation.
- Section [2.4](#) considers security aspects of the implementation.

2.1 Normative Variations

The following subsections describe normative variations from the MUST requirements of [\[W3C-DOM4\]](#).

2.1.1 [W3C-DOM4] Section 3.2 Interface Event

V0001: The NONE const is not supported

The specification states:

```
3.2 Interface Event
...
interface Event {
    ...
    const unsigned short NONE = 0;
    ...
};
```

All document modes (Internet Explorer 11)

The NONE const is not supported.

V0002: The timeStamp attribute is of type unsigned long long, not DOMTimeStamp

The specification states:

```
3.2 Interface Event
...
interface Event {
    ...
    readonly attribute DOMTimeStamp timeStamp;
    ...
};
```

IE11 Mode (Internet Explorer 11)

The timeStamp attribute is of type unsigned long long, not DOMTimeStamp:

```
readonly attribute unsigned long long timeStamp;
```

V0003: The isTrusted attribute is not unforgeable

The specification states:

```
3.2 Interface Event
...
interface Event {
    ...
    [Unforgeable] readonly attribute boolean isTrusted;
    ...
};
```

IE11 Mode (Internet Explorer 11)

The `isTrusted` attribute is not unforgeable:

```
readonly attribute boolean isTrusted;
```

V0004: The `defaultPrevented` flag is not unset when an event is initialized

The specification states:

```
3.2 Interface Event
...
To initialize an event, with type, bubbles, and cancelable, run these steps:
1. Set the initialized flag.
2. Unset the stop propagation flag, stop immediate propagation flag, and canceled
   flag.
3. Set the isTrusted attribute to false.
4. Set the target attribute to null.
5. Set the type attribute to type.
6. Set the bubbles attribute to bubbles.
7. Set the cancelable attribute to cancelable.
```

All document modes (All versions)

The `defaultPrevented` flag is not unset when an event is initialized: step 2 does not unset the flag when the `canceled` flag is unset.

V0005: The stop propagation flag is not unset when an event is initialized

The specification states:

```
3.2 Interface Event
...
To initialize an event, with type, bubbles, and cancelable, run these steps:
1. Set the initialized flag.
2. Unset the stop propagation flag, stop immediate propagation flag, and canceled
   flag.
3. Set the isTrusted attribute to false.
4. Set the target attribute to null.
5. Set the type attribute to type.
6. Set the bubbles attribute to bubbles.
7. Set the cancelable attribute to cancelable.
```

All document modes (All versions)

The stop propagation flag is not unset when an event is initialized.

2.1.2 [W3C-DOM4] Section 3.6 Interface EventTarget

V0006: When `dispatchEvent` is passed a null value it throws an `InvalidArgument` error

The specification states:

```
3.6 Interface EventTarget
...
interface EventTarget {
    ...
    boolean dispatchEvent(Event event);
};
```

All document modes (All versions)

When `dispatchEvent` is passed a null value it throws an `InvalidArgument` error instead of `TypeError`.

2.1.3 [W3C-DOM4] Section 4.2.2 Interface NonElementParentNode

V0007: The `getElementById` method returns `Element`, not nullable `Element`

The specification states:

```
4.2.2 Interface NonElementParentNode
...
interface NonElementParentNode {
    Element? getElementById(DOMString elementId);
};
```

All document modes (All versions)

The `getElementById` method returns `Element`, not nullable `Element`:

```
Element getElementById(DOMString elementId);
```

V0008: The `DocumentFragment` interface does not implement the `NonElementParentNode` interface

The specification states:

```
4.2.2 Interface NonElementParentNode
...
interface NonElementParentNode {
    ...
};
...
DocumentFragment implements NonElementParentNode;
```

All document modes (All versions)

The `DocumentFragment` interface does not implement the `NonElementParentNode` interface.

2.1.4 [W3C-DOM4] Section 4.2.3 Interface ParentNode

V0009: The Document and DocumentFragment interfaces do not implement the firstElementChild, lastElementChild, and childElementCount attributes

The specification states:

```
4.2.3 Interface ParentNode
...
interface ParentNode {
    ...
    readonly attribute Element? firstElementChild;
    readonly attribute Element? lastElementChild;
    readonly attribute unsigned long childElementCount;
    ...
};
Document implements ParentNode;
DocumentFragment implements ParentNode;
```

All document modes (All versions)

The Document and DocumentFragment interfaces do not implement the firstElementChild, lastElementChild, and childElementCount attributes.

V0010: The firstElementChild and lastElementChild attributes are of type Element, not nullable Element

The specification states:

```
4.2.3 Interface ParentNode
...
interface ParentNode {
    ...
    readonly attribute Element? firstElementChild;
    readonly attribute Element? lastElementChild;
    ...
};
```

All document modes (All versions)

The firstElementChild and lastElementChild attributes are of type Element, not nullable Element:

```
readonly attribute Element firstElementChild;
readonly attribute Element lastElementChild;
```

V0011: The children attribute is not correctly implemented in the Document, DocumentFragment, and Element interfaces

The specification states:

```
4.2.3 Interface ParentNode
...
interface ParentNode {
```

```
[SameObject] readonly attribute HTMLCollection children;
...
};
Element implements ParentNode;
Document implements ParentNode;
DocumentFragment implements ParentNode;
```

All document modes (All versions)

The `children` attribute is not correctly implemented in the `Document`, `DocumentFragment`, and `Element` interfaces.

V0012: The `querySelector` function is of type `Element`, not nullable `Element`

The specification states:

```
4.2.3 Interface ParentNode
...
interface ParentNode {
...
    Element? querySelector(DOMString selectors);
...
};
```

All document modes (All versions)

The `querySelector` function is of type `Element`, not nullable `Element`:

```
Element querySelector(DOMString selectors);
```

V0013: The `querySelectorAll` function does not define itself as `[NewObject]`

The specification states:

```
4.2.3 Interface ParentNode
...
interface ParentNode {
...
    [NewObject] NodeList querySelectorAll(DOMString selectors);
};
```

All document modes (All versions)

The `querySelectorAll` function does not define itself as `[NewObject]`:

```
NodeList querySelectorAll(DOMString selectors);
```

2.1.5 [W3C-DOM4] Section 4.2.4 Interface NonDocumentTypeChildNode

V0014: The `previousElementSibling` and `nextElementSibling` attributes are of type `Element`, not nullable `Element`

The specification states:

```

4.2.4 Interface NonDocumentTypeChildNode
...
interface NonDocumentTypeChildNode {
    readonly attribute Element? previousElementSibling;
    readonly attribute Element? nextElementSibling;
};

```

IE11 Mode (Internet Explorer 11)

The `previousElementSibling` and `nextElementSibling` attributes are of type `Element`, not nullable `Element`:

```

    readonly attribute Element previousElementSibling;
    readonly attribute Element nextElementSibling;

```

V0015: The `CharacterData` interface does not implement the `NonDocumentTypeChildNode` interface

The specification states:

```

4.2.4 Interface NonDocumentTypeChildNode
...
interface NonDocumentTypeChildNode {
    ...
};
...
CharacterData implements NonDocumentTypeChildNode;

```

All document modes (All versions)

The `CharacterData` interface does not implement the `NonDocumentTypeChildNode` interface.

2.1.6 [W3C-DOM4] Section 4.5.1 Interface DOMImplementation

V0016: The `hasFeature` method takes two arguments

The specification states:

```

4.5.1 Interface DOMImplementation
User agents must create a DOMImplementation object whenever a document is created and
associate it with that document.
...
interface DOMImplementation {
    ...
    boolean hasFeature(); // useless; always returns true
};

```

IE11 Mode (All versions)

The `hasFeature` method takes two arguments, not zero:

```

    boolean hasFeature(DOMString? feature, DOMString? version);

```

2.1.7 [W3C-DOM4] Section 4.8 Interface Element

V0017: The namespaceURI, and localName attributes, and the attributes attribute, are inherited from the Node interface

The specification states:

```
4.8 Interface Element
...
interface Element : Node {
    readonly attribute DOMString? namespaceURI;
    ...
    readonly attribute DOMString localName;
    ...
    [SameObject] readonly attribute NamedNodeMap attributes;
    ...
};
```

All document modes (All versions)

The namespaceURI, and localName attributes, and the attributes attribute, are inherited from the Node interface instead of being defined on the Element interface.

V0019: The getElementsByTagName method is not supported on the Element instance

The specification states:

```
4.8 Interface Element
...
interface Element : Node {
    ...
    HTMLCollection getElementsByTagName(DOMString classNames);
};
```

All document modes (All versions)

The getElementsByTagName method is not supported on the Element instance (however, it is available on the HTMLCollection interface).

2.1.8 [W3C-DOM4] Section 4.8.1 Interface Attr

V0020: The namespaceURI and localName attributes are not supported

The specification states:

```
4.8.1 Interface Attr
...
interface Attr {
    readonly attribute DOMString? namespaceURI;
    ...
    readonly attribute DOMString localName;
    ...
};
```

All document modes (All versions)

The `namespaceURI` and `localName` attributes are not supported.

2.1.9 [W3C-DOM4] Section 4.9 Interface `CharacterData`

V0021: The `data` attribute when set to null returns null, not `EmptyString`

The specification states:

```
4.9 Interface CharacterData
...
interface CharacterData : Node {
    [TreatNullAs=EmptyString] attribute DOMString data;
    ...
};
```

All document modes (All versions)

The `data` attribute when set to null returns null, not `EmptyString`.

V0022: The `deleteData` and `replaceData` methods throw an `IndexSizeError` when passed negative values

The specification states:

```
4.9 Interface CharacterData
...
interface CharacterData : Node {
    ...
    void deleteData(unsigned long offset, unsigned long count);
    void replaceData(unsigned long offset, unsigned long count, DOMString data);
};
```

All document modes (All versions)

The `deleteData` and `replaceData` methods throw an `IndexSizeError` when passed negative values. They should instead treat negative values as if they were zero.

2.1.10 [W3C-DOM4] Section 4.10 Interface `Text`

V0023: The `Text` constructor is not supported

The specification states:

```
4.10 Interface Text
...
[Constructor(optional DOMString data = ""),
 Exposed=Window]
interface Text : CharacterData {
    ...
};
```

All document modes (All versions)

The `Text` constructor is not supported.

2.1.11 [W3C-DOM4] Section 5.2 Interface Range

V0024: The `isPointInRange`, `comparePoint`, and `intersectsNode` methods are not supported

The specification states:

```
5.2 Interface Range
...
interface Range {
  ...
  boolean isPointInRange(Node node, unsigned long offset);
  short comparePoint(Node node, unsigned long offset);

  boolean intersectsNode(Node node);
  ...
};
```

All document modes (All versions)

The `isPointInRange`, `comparePoint`, and `intersectsNode` methods are not supported.

2.1.12 [W3C-DOM4] Section 6.1 Interface NodeIterator

V0025: The `referenceNode` and `pointerBeforeReferenceNode` attributes are not supported

The specification states:

```
6.1 Interface NodeIterator
...
interface NodeIterator {
  ...
  readonly attribute Node referenceNode;
  readonly attribute boolean pointerBeforeReferenceNode;
  ...
};
```

All document modes (All versions)

The `referenceNode` and `pointerBeforeReferenceNode` attributes are not supported.

2.2 Clarifications

There are no clarifications of the MAY and SHOULD requirements of [\[W3C-DOM4\]](#).

2.3 Error Handling

There are no additional error handling considerations.

2.4 Security

There are no additional security considerations.

3 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

4 Index

C

[Change tracking](#) 16

G

[Glossary](#) 4

I

[Informative references](#) 4

[Introduction](#) 4

N

[Normative references](#) 4

R

References

[informative](#) 4

[normative](#) 4

T

[Tracking changes](#) 16