

## [MS-WEBIDL1]:

# Microsoft Edge / Internet Explorer WebIDL Level 1 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](mailto: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](http://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](mailto:dochelp@microsoft.com).

## Revision Summary

| Date      | Revision History | Revision Class | Comments   |
|-----------|------------------|----------------|--|
| 4/25/2017 | 1.0              | New            | Released new document.   |
| 10/3/2017 | 1.0              | None           | No changes to the meaning, language, or formatting of the technical content. |
| 2/22/2018 | 1.0              | None           | No changes to the meaning, language, or formatting of the technical content. |
| 3/23/2018 | 1.0              | None           | No changes to the meaning, language, or formatting of the technical content. |
| 8/28/2018 | 1.0              | None           | No changes to the meaning, language, or formatting of the technical content. |

# Table of Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction .....</b>  | <b>4</b>  |
| 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.....  | 6         |
| <b>2</b> | <b>Standards Support Statements.....</b>                                 | <b>7</b>  |
| 2.1      | Normative Variations .....   | 7         |
| 2.1.1    | [W3C-WEBIDL1] Section 3.2.7 Iterable declarations .....                  | 7         |
| 2.1.2    | [W3C-WEBIDL1] Section 3.10.17 USVString .....                            | 8         |
| 2.1.3    | [W3C-WEBIDL1] Section 4.2.18 USVString .....                             | 8         |
| 2.1.4    | [W3C-WEBIDL1] Section 4.2.25.1 Creating a sequence from an iterable..... | 8         |
| 2.1.5    | [W3C-WEBIDL1] Section 4.6.3 Dictionary constructors.....                 | 9         |
| 2.1.6    | [W3C-WEBIDL1] Section 4.6.9 Common iterator behavior .....               | 9         |
| 2.1.7    | [W3C-WEBIDL1] Section 4.6.9.1 @@iterator .....                           | 10        |
| 2.1.8    | [W3C-WEBIDL1] Section 4.6.9.2 forEach .....                              | 10        |
| 2.1.9    | [W3C-WEBIDL1] Section 4.6.10 Iterable declarations.....                  | 11        |
| 2.1.10   | [W3C-WEBIDL1] Section 4.6.10.1 entries .....                             | 12        |
| 2.1.11   | [W3C-WEBIDL1] Section 4.6.10.2 keys .....                                | 12        |
| 2.1.12   | [W3C-WEBIDL1] Section 4.6.10.3 values.....                               | 13        |
| 2.1.13   | [W3C-WEBIDL1] Section 4.6.10.4 Default iterator objects .....            | 14        |
| 2.1.14   | [W3C-WEBIDL1] Section 4.6.10.5 Iterator prototype object.....            | 14        |
| 2.1.15   | [W3C-WEBIDL1] Section 4.6.11 Initializing objects from iterables .....   | 15        |
| 2.2      | Clarifications .....   | 16        |
| 2.2.1    | [W3C-WEBIDL1] Section 3.10.22 Callback function types .....              | 16        |
| 2.3      | Extensions .....   | 16        |
| 2.4      | Error Handling .....   | 16        |
| 2.5      | Security .....   | 17        |
| <b>3</b> | <b>Change Tracking.....</b>  | <b>18</b> |
| <b>4</b> | <b>Index.....</b>  | <b>19</b> |

# 1 Introduction

This document describes the level of support provided by Microsoft web browsers for the *WebIDL Level 1* specification [\[W3C-WEBIDL1\]](#), published 15 December 2016. The [W3C-WEBIDL1] specification defines an interface definition language, Web IDL, that can be used to describe interfaces that are intended to be implemented in web browsers. Web IDL is an IDL variant with a number of features that allow the behavior of common script objects in the web platform to be specified more readily.

## 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](mailto: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-WEBIDL1] World Wide Web Consortium, "WebIDL Level 1", W3C Recommendation 15 December 2016, <https://www.w3.org/TR/2016/REC-WebIDL-1-20161215/>

### 1.2.2 Informative References

None.

## 1.3 Microsoft Implementations

The following Microsoft web browser versions implement some portion of the [\[W3C-WEBIDL1\]](#) specification:

- Windows Internet Explorer 9
- Windows Internet Explorer 10
- Internet Explorer 11
- Internet Explorer 11 for Windows 10
- 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  |
|-------------------------------------|---|
| Internet Explorer 9                 | Quirks Mode<br>IE7 Mode<br>IE8 Mode<br>IE9 Mode                           |
| Internet Explorer 10                | Quirks Mode<br>IE7 Mode<br>IE8 Mode<br>IE9 Mode<br>IE10 Mode              |
| Internet Explorer 11                | Quirks Mode<br>IE7 Mode<br>IE8 Mode<br>IE9 Mode<br>IE10 Mode<br>IE11 Mode |
| Internet Explorer 11 for Windows 10 | Quirks Mode<br>IE7 Mode<br>IE8 Mode<br>IE9 Mode<br>IE10 Mode<br>IE11 Mode |
| 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 [\[W3C-WEBIDL1\]](#), 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 [\[W3C-WEBIDL1\]](#) and whether they are considered normative or informative.

| Sections                      | Normative/Informative |
|-------------------------------|-----------------------|
| 1                             | Informative           |
| 2                             | Normative             |
| 6-8                           | Informative           |
| Appendices A, B, B.1, and B.2 | Normative             |

## 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 <a href="#">RFC2119</a> .) 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, clarifications, and extensions for the Microsoft implementation of [\[W3C-WEBIDL1\]](#).

- Section [2.1](#) describes normative variations from the MUST requirements of the specification.
- Section [2.2](#) describes clarifications of the MAY and SHOULD requirements.
- Section [2.3](#) describes extensions to the requirements.
- Section [2.4](#) considers error handling aspects of the implementation.
- Section [2.5](#) considers security aspects of the implementation.

### 2.1 Normative Variations

The following subsections describe normative variations from the MUST requirements of [\[W3C-WEBIDL1\]](#).

#### 2.1.1 [W3C-WEBIDL1] Section 3.2.7 Iterable declarations

V0001: Iterable declarations are not supported

The specification states:

3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

4.2.25.1 Creating a sequence from an iterable

4.6.9 Common iterator behavior

4.6.9.1 `@@iterator`

4.6.9.2 `forEach`

4.6.10 Iterable declarations

4.6.10.1 `entries`

4.6.10.2 `keys`

4.6.10.3 `values`

4.6.10.4 Default iterator objects

4.6.10.5 Iterator prototype object

4.6.11 Initializing objects from iterables

#### ***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

## 2.1.2 [W3C-WEBIDL1] Section 3.10.17 USVString

V0002: The USVString type is not supported

The specification states:

3.10.17 USVString

The USVString type corresponds to the set of all possible sequences of Unicode scalar values, which are all of the Unicode code points apart from the surrogate code points.

### ***IE11 Mode, IE10 Mode, and IE9 Mode (All versions)***

The USVString type is not supported.

## 2.1.3 [W3C-WEBIDL1] Section 4.2.18 USVString

V0002: The USVString type is not supported

The specification states:

3.10.17 USVString

The USVString type corresponds to the set of all possible sequences of Unicode scalar values, which are all of the Unicode code points apart from the surrogate code points.

### ***IE11 Mode, IE10 Mode, and IE9 Mode (All versions)***

The USVString type is not supported.

## 2.1.4 [W3C-WEBIDL1] Section 4.2.25.1 Creating a sequence from an iterable

V0001: Iterable declarations are not supported

The specification states:

3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching Iterable) in the body of the interface.

4.2.25.1 Creating a sequence from an iterable

4.6.9 Common iterator behavior

4.6.9.1 @@iterator

4.6.9.2 forEach

4.6.10 Iterable declarations

4.6.10.1 entries

4.6.10.2 keys

- 4.6.10.3 values
- 4.6.10.4 Default iterator objects
- 4.6.10.5 Iterator prototype object
- 4.6.11 Initializing objects from iterables

### ***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

## **2.1.5 [W3C-WEBIDL1] Section 4.6.3 Dictionary constructors**

V0003: Dictionary constructors are not supported

The specification states:

### 4.6.3 Dictionary constructors

For every dictionary type that has one or more [Constructor] extended attributes and which is exposed in a given ECMAScript global environment, a corresponding property must exist on the ECMAScript environment's global object. The name of the property is the identifier of the dictionary, and its value is a function object called the dictionary constructor.

### ***All document modes (All versions)***

Dictionary constructors are not supported.

## **2.1.6 [W3C-WEBIDL1] Section 4.6.9 Common iterator behavior**

V0001: Iterable declarations are not supported

The specification states:

### 3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

### 4.2.25.1 Creating a sequence from an iterable

### 4.6.9 Common iterator behavior

#### 4.6.9.1 @@iterator

#### 4.6.9.2 `forEach`

### 4.6.10 Iterable declarations

#### 4.6.10.1 `entries`

#### 4.6.10.2 `keys`

#### 4.6.10.3 `values`

- 4.6.10.4 Default iterator objects
- 4.6.10.5 Iterator prototype object
- 4.6.11 Initializing objects from iterables

***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

**2.1.7 [W3C-WEBIDL1] Section 4.6.9.1 @@iterator**

V0001: Iterable declarations are not supported

The specification states:

- 3.2.7 Iterable declarations
  - An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.
- 4.2.25.1 Creating a sequence from an iterable
- 4.6.9 Common iterator behavior
  - 4.6.9.1 @@iterator
  - 4.6.9.2 `forEach`
- 4.6.10 Iterable declarations
  - 4.6.10.1 `entries`
  - 4.6.10.2 `keys`
  - 4.6.10.3 `values`
  - 4.6.10.4 Default iterator objects
  - 4.6.10.5 Iterator prototype object
  - 4.6.11 Initializing objects from iterables

***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

**2.1.8 [W3C-WEBIDL1] Section 4.6.9.2 `forEach`**

V0001: Iterable declarations are not supported

The specification states:

### 3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

#### 4.2.25.1 Creating a sequence from an iterable

#### 4.6.9 Common iterator behavior

##### 4.6.9.1 `@@iterator`

##### 4.6.9.2 `forEach`

#### 4.6.10 Iterable declarations

##### 4.6.10.1 `entries`

##### 4.6.10.2 `keys`

##### 4.6.10.3 `values`

##### 4.6.10.4 Default iterator objects

##### 4.6.10.5 Iterator prototype object

##### 4.6.11 Initializing objects from iterables

## ***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

### **2.1.9 [W3C-WEBIDL1] Section 4.6.10 Iterable declarations**

V0001: Iterable declarations are not supported

The specification states:

### 3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

#### 4.2.25.1 Creating a sequence from an iterable

#### 4.6.9 Common iterator behavior

##### 4.6.9.1 `@@iterator`

##### 4.6.9.2 `forEach`

#### 4.6.10 Iterable declarations

##### 4.6.10.1 `entries`

##### 4.6.10.2 `keys`

##### 4.6.10.3 `values`

##### 4.6.10.4 Default iterator objects

##### 4.6.10.5 Iterator prototype object

4.6.11 Initializing objects from iterables

***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

**2.1.10 [W3C-WEBIDL1] Section 4.6.10.1 entries**

V0001: Iterable declarations are not supported

The specification states:

3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

4.2.25.1 Creating a sequence from an iterable

4.6.9 Common iterator behavior

4.6.9.1 `@@iterator`

4.6.9.2 `forEach`

4.6.10 Iterable declarations

4.6.10.1 entries

4.6.10.2 keys

4.6.10.3 values

4.6.10.4 Default iterator objects

4.6.10.5 Iterator prototype object

4.6.11 Initializing objects from iterables

***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

**2.1.11 [W3C-WEBIDL1] Section 4.6.10.2 keys**

V0001: Iterable declarations are not supported

The specification states:

3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

- 4.2.25.1 Creating a sequence from an iterable
- 4.6.9 Common iterator behavior
  - 4.6.9.1 @@iterator
  - 4.6.9.2 forEach
- 4.6.10 Iterable declarations
  - 4.6.10.1 entries
  - 4.6.10.2 keys
  - 4.6.10.3 values
  - 4.6.10.4 Default iterator objects
  - 4.6.10.5 Iterator prototype object
- 4.6.11 Initializing objects from iterables

***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

**2.1.12 [W3C-WEBIDL1] Section 4.6.10.3 values**

V0001: Iterable declarations are not supported

The specification states:

- 3.2.7 Iterable declarations
  - An interface can be declared to be iterable by using an iterable declaration (matching Iterable) in the body of the interface.
- 4.2.25.1 Creating a sequence from an iterable
- 4.6.9 Common iterator behavior
  - 4.6.9.1 @@iterator
  - 4.6.9.2 forEach
- 4.6.10 Iterable declarations
  - 4.6.10.1 entries
  - 4.6.10.2 keys
  - 4.6.10.3 values
  - 4.6.10.4 Default iterator objects
  - 4.6.10.5 Iterator prototype object
- 4.6.11 Initializing objects from iterables

**IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)**

Iterable declarations are not supported.

**2.1.13 [W3C-WEBIDL1] Section 4.6.10.4 Default iterator objects**

V0001: Iterable declarations are not supported

The specification states:

3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

4.2.25.1 Creating a sequence from an iterable

4.6.9 Common iterator behavior

4.6.9.1 `@@iterator`

4.6.9.2 `forEach`

4.6.10 Iterable declarations

4.6.10.1 `entries`

4.6.10.2 `keys`

4.6.10.3 `values`

4.6.10.4 Default iterator objects

4.6.10.5 Iterator prototype object

4.6.11 Initializing objects from iterables

**IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)**

Iterable declarations are not supported.

**2.1.14 [W3C-WEBIDL1] Section 4.6.10.5 Iterator prototype object**

V0001: Iterable declarations are not supported

The specification states:

3.2.7 Iterable declarations

An interface can be declared to be iterable by using an iterable declaration (matching `Iterable`) in the body of the interface.

4.2.25.1 Creating a sequence from an iterable

4.6.9 Common iterator behavior

- 4.6.9.1 @@iterator
- 4.6.9.2 forEach
- 4.6.10 Iterable declarations
  - 4.6.10.1 entries
  - 4.6.10.2 keys
  - 4.6.10.3 values
  - 4.6.10.4 Default iterator objects
  - 4.6.10.5 Iterator prototype object
- 4.6.11 Initializing objects from iterables

***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

**2.1.15 [W3C-WEBIDL1] Section 4.6.11 Initializing objects from iterables**

V0001: Iterable declarations are not supported

The specification states:

- 3.2.7 Iterable declarations
  - An interface can be declared to be iterable by using an iterable declaration (matching Iterable) in the body of the interface.
- 4.2.25.1 Creating a sequence from an iterable
- 4.6.9 Common iterator behavior
  - 4.6.9.1 @@iterator
  - 4.6.9.2 forEach
- 4.6.10 Iterable declarations
  - 4.6.10.1 entries
  - 4.6.10.2 keys
  - 4.6.10.3 values
  - 4.6.10.4 Default iterator objects
  - 4.6.10.5 Iterator prototype object
- 4.6.11 Initializing objects from iterables

***IE11 Mode, IE10 Mode, IE9 Mode, IE8 Mode, IE7 Mode, and IE5 (Quirks) Mode (All versions)***

Iterable declarations are not supported.

## 2.2 Clarifications

The following subsections describe clarifications of the MAY and SHOULD requirements of [\[W3C-WEBIDL1\]](#).

### 2.2.1 [W3C-WEBIDL1] Section 3.10.22 Callback function types

C0001: No `TypeError` exception is thrown when an invalid callback function is passed to the `getCurrentPosition` method or `watchPosition` method

The specification states:

#### 3.10.22 Callback function types

An identifier that identifies a callback function is used to refer to a type whose values are references to objects that are functions with the given signature.

#### 4.2.23 Callback function types

...

An ECMAScript value *V* is converted to an IDL callback function type value by running the following algorithm:

1. If the result of calling `IsCallable(V)` is false and the conversion to an IDL value is not being performed due to *V* being assigned to an attribute whose type is a nullable callback function that is annotated with `[TreatNonObjectAsNull]`, then throw a `TypeError`.
2. Return the IDL callback function type value that represents a reference to the same object that *V* represents, with the incumbent script as the callback context. `[HTML5]`.

The result of converting an IDL callback function type value to an ECMAScript value is a reference to the same object that the IDL callback function type value represents.

### **IE11 Mode, IE10 Mode, and IE9 Mode (All versions)**

No `TypeError` exception is thrown when an invalid callback function is passed to the `getCurrentPosition` method or `watchPosition` method; instead an `Error` object is thrown:

- If the value for the callback function is null, Internet Explorer 9 throws a `DOMException` with the code attribute set to `TYPE_MISMATCH_ERR`. Internet Explorer 10, Internet Explorer 11, and Internet Explorer 11 for Windows 10 throw an `Error` object with the `number` attribute set to -2147418113.
- If the value for the callback function is not null and not a function, then Internet Explorer 9, Internet Explorer 10, Internet Explorer 11, and Internet Explorer 11 for Windows 10 throw an `Error` object with the `number` attribute set to -2147024809.

## 2.3 Extensions

There are no extensions to the requirements of [\[W3C-WEBIDL1\]](#).

## 2.4 Error Handling

There are no additional error handling considerations.

## 2.5 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](#) 18

### G

[Glossary](#) 4

### I

[Informative references](#) 4

[Introduction](#) 4

### N

[Normative references](#) 4

### R

References

[informative](#) 4

[normative](#) 4

### T

[Tracking changes](#) 18