# [MS-HTML52]:

# Microsoft Edge HTML 5.2 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 |
|---|---|---|---|
| 6/5/2018 | 1.0 | New | Released new document. |

# Table of Contents

# 1 Introduction

This document describes the level of support provided by Microsoft Edge for the *HTML 5.2* specification, [W3C-HTML52], published 14 December 2017.

The specification defines the second minor revision of the fifth major version of the Hypertext Markup Language (HTML), which is the standard markup language of the World Wide Web.

## 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-HTML52] World Wide Web Consortium, "HTML 5.2", W3C Recommendation, 14 December 2017, https://www.w3.org/TR/2017/REC-html52-20171214/

### 1.2.2 Informative References

None.

## 1.3 Microsoft Implementations

The following Microsoft web browsers implement some portion of the [W3C-HTML52] 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-HTML52] , 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-HTML52] and whether they are considered normative or informative.

| Sections | Normative/Informative |
|---|---|
| 1 | Informative |
| 2-11 | Normative |
| 12 | 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, clarifications, and extensions for the Microsoft implementation of [W3C-HTML52].

- 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-HTML52].

### 2.1.1 [W3C-HTML52] Section 2.4.1. Common parser idioms

V0001: The white space character definitions do not include all the code points with Unicode property "White_Space"

The specification states:

```
2.4.1 Common parser idioms
    ...
    The White_Space characters are those that have the Unicode property "White_Space" in
    the Unicode PropList.txt data file. [UNICODE]
```

***EdgeHTML Mode***

The white space character definitions do not include all the code points with Unicode property "White_Space" listed in the Unicode `PropList.txt` file.

### 2.1.2 [W3C-HTML52] Section 2.4.4.3. Floating-point numbers

V0374: Characters "d" and "D" can be used in place of "e" and "E" in a floating-point number

The specification states:

```
2.4.4.3 Floating-point numbers

    A string is a valid floating-point number if it consists of:
        1. Optionally, a U+002D HYPHEN-MINUS character (-).
        2. One or both of the following, in the given order:
            1. A series of one or more ASCII digits.
            2.  Both of the following, in the given order:
                1. A single U+002E FULL STOP character (.).
                2. A series of one or more ASCII digits.
        3. Optionally:
            1.  Either a U+0065 LATIN SMALL LETTER E character (e) or a U+0045 LATIN
                CAPITAL LETTER E character (E).
            2.  Optionally, a U+002D HYPHEN-MINUS character (-) or U+002B PLUS SIGN
                character (+).
```

```
3.  A series of one or more ASCII digits.
```

***EdgeHTML Mode***

Characters "d" and "D" can be used in place of "e" and "E" in a floating-point number.

### 2.1.3 [W3C-HTML52] Section 2.7.2.2. The HTMLFormControlsCollection interface

V0375: The namedItem function does not return a RadioNodeList

The specification states:

```
2.7.2.2. The HTMLFormControlsCollection interface

    The HTMLFormControlsCollection interface is used for collections of listed elements
    in form elements.

        interface HTMLFormControlsCollection : HTMLCollection {
            // inherits length and item()
            getter (RadioNodeList or Element)? namedItem(DOMString name);
                // shadows inherited namedItem()
};
```

***EdgeHTML Mode***

The `namedItem` function does not return a `RadioNodeList`; instead it returns an `HTMLCollection`.

getter (HTMLCollection or Element)? namedItem(DOMString name);

V0376: The RadioNodeList interface is not supported

The specification states:

```
2.7.2.2. The HTMLFormControlsCollection interface

    The HTMLFormControlsCollection interface is used for collections of listed elements
    in form elements.
        ...
        interface RadioNodeList : NodeList {
            attribute DOMString value;
        };
```

***EdgeHTML Mode***

The `RadioNodeList` interface is not supported.

### 2.1.4 [W3C-HTML52] Section 2.9.3. StructuredSerializeInternal ( value, forStorage [ , memory ] )

V0385: StructuredSerializeInternal is not supported

The specification states:

```
2.9.3. StructuredSerializeInternal ( value, forStorage [ , memory ] )
The StructuredSerializeInternal abstract operation takes as input a JavaScript value
value and serializes it to a Realm-independent form, represented here as a Record. This
serialized form has all the information necessary to later deserialize into a new
JavaScript value in a different Realm.
...
```

***EdgeHTML Mode***

StructuredSerializeInternal is not supported.

### 2.1.5 [W3C-HTML52] Section 2.9.6. StructuredDeserialize ( serialized, targetRealm [ , memory ] )

V0386: StructuredDeserialize is not supported

The specification states:

```
2.9.6. StructuredDeserialize ( serialized, targetRealm [ , memory ] )
The StructuredDeserialize abstract operation takes as input a Record serialized, which
was previously produced by StructuredSerialize or StructuredSerializeForStorage, and
deserializes it into a new JavaScript value, created in targetRealm.
```

***EdgeHTML Mode***

StructuredDeserialize is not supported

### 2.1.6 [W3C-HTML52] Section 2.9.7. StructuredSerializeWithTransfer ( value, transferList )

V0387: StructuredSerializeWithTransfer is not supported

The specification states:

```
2.9.7. StructuredSerializeWithTransfer ( value, transferList )
```

***EdgeHTML Mode***

StructuredSerializeWithTransfer is not supported.

### 2.1.7 [W3C-HTML52] Section 2.9.8. StructuredDeserializeWithTransfer ( serializeWithTransferResult, targetRealm )

V0388: StructuredDeserializeWithTransfer is not supported

The specification states:

```
2.9.8. StructuredDeserializeWithTransfer ( serializeWithTransferResult, targetRealm )
```

***EdgeHTML Mode***

StructuredDeserializeWithTransfer is not supported.

### 2.1.8 [W3C-HTML52] Section 3.1.1. The Document object

V0384: The open function is not defined as two separate methods

The specification states:

```
3.1.1. The Document object

    The DOM specification defines a Document interface, which this specification extends
    significantly:
        ...
        [OverrideBuiltins]
        partial interface Document {
            ...
            // dynamic markup insertion
            Document open(optional DOMString type = "text/html", optional DOMString
            replace = "");
            WindowProxy open(DOMString url, DOMString name, DOMString features,
              optional boolean replace = false);
            ...
        };
```

***EdgeHTML Mode***

The `open` function is not defined as two separate methods.

(Document or Window) open(optional DOMString url = "text/html", optional DOMString name,

optional DOMString features, optional boolean replace);

### 2.1.9 [W3C-HTML52] Section 3.2.2. Elements in the DOM

V0017: The translate attribute is not supported

The specification states:

```
3.2.2 Elements in the DOM
    ...
        interface HTMLElement : Element {
            ...
            attribute boolean translate;
            ...
        };
```

***EdgeHTML Mode***

The `translate` attribute is not supported.

V0018: The tabindex attribute is type short, not long

The specification states:

```
3.2.2 Elements in the DOM
    ...
        interface HTMLElement : Element {
            ...
            attribute long tabIndex;
            ...
        };
```

***EdgeHTML Mode***

The `tabindex` attribute is type `short`, not `long`.

V0389: The forceSpellCheck method is not supported

The specification states:

```
3.2.2. Elements in the DOM
    ...
        [HTMLConstructor]
        interface HTMLElement : Element {
            ...
            void forceSpellCheck();
            ...
        };
```

***EdgeHTML Mode***

The `forceSpellCheck` method is not supported.

### 2.1.10 [W3C-HTML52] Section 3.2.5.3. The translate attribute

V0020: The translate attribute is not supported

The specification states:

```
... The translate attribute

    The translate attribute is an enumerated attribute that is used to specify whether an
    element's attribute values and the values of its Text node children are to be
    translated when the page is localized, or whether to leave them unchanged.
```

***EdgeHTML Mode***

The `translate` attribute is not supported.

### 2.1.11 [W3C-HTML52] Section 3.2.5.5. The dir attribute

V0021: The auto keyword is not supported

The specification states:

```
... The dir attribute

    The dir attribute specifies the element's text directionality. The attribute is an
    enumerated attribute with the following keywords and states:
        ...
        The auto keyword, which maps to the auto state

            Indicates that the contents of the element are explicitly directionally
            isolated text, but that the direction is to be determined programmatically
            using the contents of the element (as described below).
```

***EdgeHTML Mode***

The `auto` keyword is not supported.

## 2.1.12 [W3C-HTML52] Section 3.2.5.7. Embedding custom non-visible data with the data-* attributes

V0022: A data- attribute that contains an uppercase letter after a dash does not insert a dash before the character

The specification states:

```
3.2.5.7 Embedding custom non-visible data with the data-* attributes
    ...
    To set the value of a new named property or set the value of an existing named
    property for a DOMStringMap, given a property name name and a new value value, run
    the following steps:
        ...
        2.  For each uppercase ASCII letter in name, insert a U+002D HYPHEN-MINUS
            character (-) before the character and replace the character with the same
            character converted to ASCII lowercase.
```

***EdgeHTML Mode***

A `data-` attribute that contains an uppercase letter after a dash does not insert a dash before the character.

V0023: A SyntaxError is not thrown when setting a data- attribute that contains a dash in the name

The specification states:

```
3.2.5.7 Embedding custom non-visible data with the data-* attributes
    ...
    To set the value of a new named property or set the value of an existing named
    property for a DOMStringMap, given a property name name and a new value value, run
    the following steps:        ...
        1. If name contains a U+002D HYPHEN-MINUS character (-) followed by a lowercase
           ASCII letter, then throw a "SyntaxError" DOMException and abort these steps.
```

***EdgeHTML Mode***

A `SyntaxError` is not thrown when setting a `data-` attribute that contains a dash in the name (e.g. data-to-string).

### 2.1.13 [W3C-HTML52] Section 3.2.6. The innerText IDL attribute

V0390: Trailing white space is not always trimmed at the end of a line or before a new line

The specification states:

```
3.2.6. The innerText IDL attribute
    …
    On getting, the innerText attribute must follow these steps:
        …
        2. Compute a list of items each of which is a string or a positive integer (a
           required line break count), by applying the following recursive procedure to
           each child node node of this element in tree order, and then concatenating the
           results to a single list of items.
            …
            4. If node is a Text node, then for each CSS text box produced by node, in
               content order, compute the text of the box after application of the CSS
               white-space processing rules and text-transform rules, let the result of
               these substeps be a list of the resulting strings, and abort these
               substeps. The CSS white-space processing rules are slightly modified:
               collapsible spaces at the end of lines are always collapsed, but they are
               only removed if the line is the last line of the block, or it ends with a
               br element. Soft hyphens should be preserved. [CSS-TEXT-3]
```

#### *EdgeHTML Mode*

Trailing white space is not always trimmed at the end of a line or before a new line.

V0391: CSS values affecting innerText are not applied

The specification states:

```
3.2.6. The innerText IDL attribute

    On getting, the [InlineCode|innerText] attribute must follow these steps:
        1. If this element is not being rendered, or if the user agent is a non-CSS user
           agent, then return the same value as the textContent IDL attribute on this
           element.
        ...
        6. Return the concatenation of the string items.
```

#### *EdgeHTML Mode*

CSS values affecting `innerText` are not applied.

### 2.1.14 [W3C-HTML52] Section 4.2.2. The title element

V0024: The directionality set in the title element does not affect the title used in the window tab

The specification states:

```
4.2.2 The title element
    ...
    User agents should use the document's title when referring to the document in their
```

```
user interface. When the contents of a title element are used in this way, the
directionality of that title element should be used to set the directionality of the
document's title in the user interface.
```

***EdgeHTML Mode***

The directionality set in the `title` element does not affect the title used in the window tab.

V0025: The text attribute of the title element does not remove leading and trailing white space from the returned string.

The specification states:

```
4.2.2 The title element
    ...
    The IDL attribute text must return ...
```

***EdgeHTML Mode***

The `text` attribute of the `title` element does not remove leading and trailing white space from the returned string.

## 2.1.15 [W3C-HTML52] Section 4.2.4. The link element

V0029: The sizes attribute is not supported

The specification states:

```
4.2.4 The link element
    ...
    Content attributes
        Global attributes
        …
        sizes — Sizes of the icons (for rel="icon")
```

***EdgeHTML Mode***

The `sizes` attribute is not supported.

## 2.1.16 [W3C-HTML52] Section 4.2.6. The style element

V0032: No error event is fired on the style element in the case of a Content-Type mismatch

The specification states:

```
4.2.6 The style element
    ...
    Once the attempts to obtain the style sheet's critical subresources, if any, are
    complete, or, if the style sheet has no critical subresources, once the style sheet
    has been parsed and processed, the user agent must, if the loads were successful or
    there were none, queue a task to fire a simple event named load at the style element,
```

```
or, if one of the style sheet's critical subresources failed to completely load for
any reason (e.g. DNS error, HTTP 404 response, a connection being prematurely closed,
unsupported Content-Type), queue a task to fire a simple event named error at the
style element.
```

***EdgeHTML Mode***

No error event is fired on the `style` element in the case of a `Content-Type` mismatch.

## 2.1.17 [W3C-HTML52] Section 4.3.1. The body element

V0033: The onerror event handler does not replace the generic event handler

The specification states:

```
4.3.1 The body element
    ...
    The onblur, onerror, onfocus, onload, onresize, and onscroll event handlers of the
    Window object, exposed on the body element, replace the generic event handlers with
    the same names normally supported by HTML elements.
```

***EdgeHTML Mode***

The `onerror` event handler does not replace the generic event handler.

## 2.1.18 [W3C-HTML52] Section 4.3.9.1. Creating an outline

V0037: There is no graphical outline mechanism

The specification states:

```
4.3.10.1 Creating an outline
    ...
    The outline for a sectioning content element or a sectioning root element consists of
    a list of one or more potentially nested sections. ...
```

***EdgeHTML Mode***

There is no graphical outline mechanism.

## 2.1.19 [W3C-HTML52] Section 4.4.6. The ol element

V0038: The reversed attribute is not supported

The specification states:

```
... The ol element
    ...
    The reversed attribute is a boolean attribute. If present, it indicates that the list
    is a descending list (..., 3, 2, 1). If the attribute is omitted, the list is an
```

```
        ascending list (1, 2, 3, ...).
```

***EdgeHTML Mode***

The `reversed` attribute is not supported.

## 2.1.20 [W3C-HTML52] Section 4.5.1. The a element

V0388: The HTMLHyperlinkElementUtils interface is not implemented for the HTMLAnchorElement interface

The specification states:

```
    4.5.1 The a element
        ...
        DOM interface:
            …
            HTMLAnchorElement implements HTMLHyperlinkElementUtils;
```

***EdgeHTML Mode***

The `HTMLHyperlinkElementUtils` interface is not implemented for the `HTMLAnchorElement` interface.

However, some `HTMLHyperlinkElementUtils` attributes are implemented on instances of `HTMLAnchorElement`. They are:

`href`

`origin`

`protocol`

`host`

`hostname`

`port`

`pathname`

`search`

`hash`

These are not implemented:

`username`

`password`

## 2.1.21 [W3C-HTML52] Section 4.5.13. The rtc element

V0045: The rtc element is not supported

The specification states:

```
... The rtc element
    ...
    The rtc element marks a ruby text container for ruby text components in a ruby
    annotation. ...
```

***EdgeHTML Mode***

The `rtc` element is not supported.

### 2.1.22 [W3C-HTML52] Section 4.5.26. The bdi element

V0046: The bdi element is not supported

The specification states:

```
4.5.26 The bdi element
    ...
    The bdi element represents a span of text that is to be isolated from its
    surroundings for the purposes of bidirectional text formatting.
```

***EdgeHTML Mode***

The `bdi` element is not supported.

### 2.1.23 [W3C-HTML52] Section 4.7.5. The img element

V0048: The first page of a PDF document is not displayed when set in the img element

The specification states:

```
... The img element
    ...
    ... User agents must only display the first page of a multipage resource ...
```

***EdgeHTML Mode***

The first page of a PDF document is not displayed when set in the `img` element.

### 2.1.24 [W3C-HTML52] Section 4.7.7. The embed element

V0050: The type attribute is not supported

The specification states:

```
... The embed element
    ...
    DOM interface:
        interface HTMLEmbedElement : HTMLElement {
            ...
            attribute DOMString type;
            ...
```

```
        };
```

***EdgeHTML Mode***

The `type` attribute is not supported.

## 2.1.25 [W3C-HTML52] Section 4.7.8. The object element

V0051: The typeMustMatch attribute is not supported

The specification states:

```
... The object element
    ...
    DOM interface:
        interface HTMLObjectElement : HTMLElement {
            ...
            attribute boolean typeMustMatch;
            ...
        };
```

***EdgeHTML Mode***

The `typeMustMatch` attribute is not supported.

## 2.1.26 [W3C-HTML52] Section 4.7.13. Media elements

V0059: The getStartDate function is not supported

The specification states:

```
... Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        // playback state
        ...
        ... getStartDate();
        ...
    };
```

***EdgeHTML Mode***

The `getStartDate` function is not supported.

V0435: The fastSeek function is not supported

The specification states:

```
4.7.13. Media elements
    HTMLMediaElement objects (audio and video, in this specification) are simply known as
    media elements.
```

```
        ...
        interface HTMLMediaElement : HTMLElement {
        ...
            void fastSeek(double time);
        …
        };
```

```
4.7.13.9. Seeking
    …
    The fastSeek() method must seek to the time given by the method's argument, with the
    approximate-for-speed flag set.
```

***EdgeHTML Mode***

The `fastSeek` function is not supported.

## 2.1.27 [W3C-HTML52] Section 4.7.13.2. Location of the media resource

V0069: The resource selection algorithm does not set currentSrc to an absolute URL

The specification states:

```
... Location of the media resource
    ...
    The currentSrc IDL attribute is initially the empty string. Its value is changed by
    the resource selection algorithm defined below.
```

***EdgeHTML Mode***

The resource selection algorithm does not set `currentSrc` to an absolute URL; the file name is missing.

## 2.1.28 [W3C-HTML52] Section 4.7.13.5. Loading the media resource

V0070: The loadstart event is not fired when a source element is added to a video element

The specification states:

```
... Loading the media resource
    ...
    The resource selection algorithm for a media element is as follows. ...
        ...
        8.  Queue a task to fire a simple event named loadstart at the media element.
```

***EdgeHTML Mode***

The `loadstart` event is not fired when a `source` element is added to a `video` element.

V0071: The suspend event is not fired when preload=none

The specification states:

```
... Loading the media resource
    ...
    The resource fetch algorithm for a media element and a given ... URL or media
    provider object is as follows:
        ...
        4.  Run the appropriate steps from the following list:
            If mode is remote
                1.  Optionally, run the following substeps. ...
                    ...
                    2.  Queue a task to fire a simple event named suspend at the element
                        ...
```

***EdgeHTML Mode***

The `suspend` event is not fired when `preload=none`.

V0075: The src attribute incorrectly resolves invalid data: URLs as valid

The specification states:

```
... Loading the media resource
    ...
    The resource selection algorithm for a media element is as follows. ...
        ...
        9.  Run the appropriate steps from the following list:
            ...
            ...  If mode is attribute ...
                ...
                3.  If [urlString/absolute URL] was obtained successfully, set the
                    currentSrc attribute to [urlString/absolute URL].
```

***EdgeHTML Mode***

The `src` attribute incorrectly resolves invalid `data:` URLs as valid.

## 2.1.29 [W3C-HTML52] Section 4.7.13.6. Offsets into the media resource

V0076: The currentTime attribute returns a negative value if readyState is HAVE_NOTHING

The specification states:

```
... Offsets into the media resource
    ...
    The currentTime attribute must, on getting, return the media element's default
    playback start position, unless that is zero, in which case it must return the
    element's official playback position. ...
```

***EdgeHTML Mode***

The `currentTime` attribute returns a negative value if `readyState` is `HAVE_NOTHING`.

### 2.1.30 [W3C-HTML52] Section 4.7.13.9. Seeking

V0435: The fastSeek function is not supported

The specification states:

```
4.7.13. Media elements
    HTMLMediaElement objects (audio and video, in this specification) are simply known as
    media elements.
        ...
        interface HTMLMediaElement : HTMLElement {
        ...
            void fastSeek(double time);
        …
        };

4.7.13.9. Seeking
    …
    The fastSeek() method must seek to the time given by the method's argument, with the
    approximate-for-speed flag set.
```

***EdgeHTML Mode***

The `fastSeek` function is not supported.

### 2.1.31 [W3C-HTML52] Section 4.7.14. The map element

V0117: The images collection is not supported

The specification states:

```
... The map element
    …
    DOM interface
        interface HTMLMapElement : HTMLElement {
            ...
            ... readonly attribute HTMLCollection images;
        };
```

***EdgeHTML Mode***

The `images` collection is not supported.

### 2.1.32 [W3C-HTML52] Section 4.7.15. The area element

V0120: The hreflang attribute is not supported

The specification states:

```
... The area element
    …
    DOM interface
        interface HTMLAreaElement : HTMLElement {
            ...
            attribute DOMString hreflang;
            ...
```

```
        };
```

***EdgeHTML Mode***

The `hreflang` attribute is not supported.

V0121: The type attribute is not supported

The specification states:

```
... The area element
    …
    DOM interface
        interface HTMLAreaElement : HTMLElement {
            ...
            attribute DOMString type;
            ...
        };
```

***EdgeHTML Mode***

The `type` attribute is not supported.

V0123: The "default" keyword is not supported for the shape attribute

The specification states:

```
... The area element
    ...
    The shape attribute is an enumerated attribute. The following table lists the
    keywords defined for this attribute. The states given in the first cell of the rows
    with keywords give the states to which those keywords map. Some of the keywords are
    non-conforming, as noted in the last column.

        State           | Keywords  | Notes
        ----------------------------------------
        Circle state    | circle    |
                        | circ      | Non-conforming
        Default state   | default   |
        Polygon state   | poly      |
                        | polygon   | Non-conforming
        Rectangle state | rect      |
                        | rectangle | Non-conforming
```

***EdgeHTML Mode***

The "default" keyword value is not supported for the `shape` attribute.

V0124: If the radius is negative, the absolute value is used

The specification states:

```
... The area element
    ...
```

```
In the circle state, area elements must have a coords attribute present, with three
integers, the last of which must be non-negative. The first integer must be the
distance in CSS pixels from the left edge of the image to the center of the circle,
the second integer must be the distance in CSS pixels from the top edge of the image
to the center of the circle, and the third integer must be the radius of the circle,
again in CSS pixels.
```

***EdgeHTML Mode***

If the radius is negative, the absolute value is used.

V0125: For the polygon state, fewer than 6 integers can be provided

The specification states:

```
... The area element
    ...
    In the polygon state, area elements must have a coords attribute with at least six
    integers, and the number of integers must be even. Each pair of integers must
    represent a coordinate given as the distances from the left and the top of the image
    in CSS pixels respectively, and all the coordinates together must represent the
    points of the polygon, in order.
```

***EdgeHTML Mode***

For the polygon state, fewer than 6 integers can be provided. If so, the missing integers are taken to be 0.

V0126: For the rectangle state, fewer than four integers can be provided

The specification states:

```
... The area element
    ...
    In the rectangle state, area elements must have a coords attribute with exactly four
    integers, the first of which must be less than the third, and the second of which
    must be less than the fourth. The four points must represent, respectively, the
    distance from the left edge of the image to the left side of the rectangle, the
    distance from the top edge to the top side, the distance from the left edge to the
    right side, and the distance from the top edge to the bottom side, all in CSS pixels.
```

***EdgeHTML Mode***

For the rectangle state, fewer than four integers can be provided. If so, the missing integers are taken to be 0.

## 2.1.33 [W3C-HTML52] Section 4.7.16.2. Processing model

V0127: If an image does not load, a valid image map will still be applied to the missing image, and not in a way that associates the image with the text

The specification states:

```
... Processing model
    ...
    If the user agent intends to show the text that the img element represents, then it
    must use the following steps.
        ...
        3.  Each remaining area element in areas represents a hyperlink. Those hyperlinks
            should all be made available to the user in a manner associated with the text
            of the img.
```

***EdgeHTML Mode***

If an image does not load, a valid image map will still be applied to the missing image, and not in a way that associates the image with the text.

V0128: The usemap attribute does not do a case-sensitive match for the appropriate image map

The specification states:

```
... Processing model

    If an img element ... has a usemap attribute specified, user agents must process it
    as follows:
```

***EdgeHTML Mode***

The `usemap` attribute does not do a case-sensitive match for the appropriate image map.

## 2.1.34 [W3C-HTML52] Section 4.7.17. MathML

V0129: The math element is not supported

The specification states:

```
... MathML

    The [MathML math/math element from the MathML namespace] falls into the embedded
    content, phrasing content, ... flow content ... categories for the purposes of the
    content models in this specification.
```

***EdgeHTML Mode***

The `math` element is not supported.

## 2.1.35 [W3C-HTML52] Section 4.9.5. The tbody element

V0133: The deleteRow function does not require the index value

The specification states:

```
4.9.5 The tbody element
    ...
    DOM interface:
```

```
interface HTMLTableSectionElement : HTMLElement {
    ...
    void deleteRow(long index);
};
```

***EdgeHTML Mode***

The `deleteRow` function does not require the `index` value.

### 2.1.36 [W3C-HTML52] Section 4.9.8. The tr element

V0135: The deleteCell method does not require the index argument

The specification states:

```
4.9.8 The tr element
    ...
    DOM interface:
        interface HTMLTableRowElement : HTMLElement {
            ...
            void deleteCell(long index);
        };
```

***EdgeHTML Mode***

The `deleteCell` method does not require the `index` argument.

### 2.1.37 [W3C-HTML52] Section 4.9.10. The th element

V0137: The abbr attribute is not defined directly on the HTMLTableHeaderCellElement interface

The specification states:

```
4.9.10 The th element
    ...
    DOM interface:
        interface HTMLTableHeaderCellElement : HTMLTableCellElement {
            ...
            attribute DOMString abbr;
        };
```

***EdgeHTML Mode***

The `abbr` attribute is not defined directly on the `HTMLTableHeaderCellElement` interface. Instead, it is abstracted to the base class `HTMLTableCellElement`.

### 2.1.38 [W3C-HTML52] Section 4.10.5. The input element

V0144: The value sanitization algorithm is not invoked when the input type attribute changes state

The specification states:

```
4.10.5 The input element
    ...
    When an input element's type attribute changes state, the user agent must run the
    following steps:
        ...
        ... Invoke the value sanitization algorithm, if one is defined for the type
        attribute's new state.
```

***EdgeHTML Mode***

The value sanitization algorithm is not invoked when the input type attribute changes state (e.g. from hidden to text).

V0145: The selection interface objects are defined, but selection does not occur on any input controls when called from script

The specification states:

```
4.10.5 The input element
    …
    DOM interface
        interface HTMLInputElement : HTMLElement {
            ...
            void select();
            attribute unsigned long selectionStart;
            attribute unsigned long selectionEnd;
            attribute DOMString selectionDirection;
            void setRangeText(DOMString replacement);
            void setRangeText(DOMString replacement, unsigned long start, unsigned long
            end, ...
            void setSelectionRange(unsigned long start, unsigned long end, optional
            DOMString direction);
        };
```

***EdgeHTML Mode***

The selection interface objects are defined, but selection does not occur on any input controls when called from script.

V0146: The dirName attribute is not supported

The specification states:

```
4.10.5 The input element
    ...
    DOM interface
        interface HTMLInputElement : HTMLElement {
            ...
            attribute DOMString dirName;
            ...
        };
```

***EdgeHTML Mode***

The `dirName` attribute is not supported.

V0149: The setRangeText functions are not supported

The specification states:

```
4.10.5 The input element
    ...
    DOM interface
        interface HTMLInputElement : HTMLElement {
            ...
            void setRangeText(DOMString replacement);
            void setRangeText(DOMString replacement, unsigned long start, unsigned long
            end, ...
            ...
        };
```

***EdgeHTML Mode***

The `setRangeText` functions are not supported.

## 2.1.39 [W3C-HTML52] Section 4.10.5.1.1. Hidden state (type=hidden)

V0150: The files attribute returns undefined, not null

The specification states:

```
4.10.5.1.1 Hidden state (type=hidden)
    …
    Bookkeeping details
        ...
        ▪ The following IDL attributes and methods do not apply to the element: checked,
        files, list, selectionStart, selectionEnd, selectionDirection, valueAsDate, and
        valueAsNumber IDL attributes; select(), setRangeText(), setSelectionRange(),
        stepDown(), and stepUp() methods.
```

***EdgeHTML Mode***

The `files` attribute returns `undefined`, not `null`.

## 2.1.40 [W3C-HTML52] Section 4.10.5.1.4. URL state (type=url)

V0151: Value sanitization does not strip leading and trailing whitespace from a URL

The specification states:

```
4.10.5.1.4 URL state (type=url)
    ...
    The value sanitization algorithm is as follows: Strip line breaks from the value,
    then strip leading and trailing whitespace from the value.
```

***EdgeHTML Mode***

Value sanitization does not strip leading and trailing whitespace from a URL.

### 2.1.41 [W3C-HTML52] Section 4.10.5.1.12. Number state (type=number)

V0156: White space in floating-point number values is treated as invalid

The specification states:

```
... Number state (type=number)
    ...
    If the element is mutable, the user agent should allow the user to change the number
    represented by its value, as obtained from applying the rules for parsing
    floating-point number values to it. ...
```

#### *EdgeHTML Mode*

White space in floating-point number values is treated as invalid; it should be ignored.

### 2.1.42 [W3C-HTML52] Section 4.10.5.1.13. Range state (type=range)

V0157: The min and max attributes allow invalid values to be specified (e.g. "AA")

The specification states:

```
... Range state (type=range)
    ...
    The min attribute, if specified, must have a value that is a valid floating-point
    number. The default minimum is 0. The max attribute, if specified, must have a value
    that is a valid floating-point number. The default maximum is 100.
```

#### *EdgeHTML Mode*

The `min` and `max` attributes allow invalid values to be specified (e.g. "AA").

V0158: The default value for the min and max attributes is the empty string ("")

The specification states:

```
... Range state (type=range)
    ...
    The min attribute, if specified, must have a value that is a valid floating-point
    number. The default minimum is 0. The max attribute, if specified, must have a value
    that is a valid floating-point number. The default maximum is 100.
```

#### *EdgeHTML Mode*

The default value for the `min` and `max` attributes is the empty string ("").

V0159: The default step is incorrect if a non-integer value is specified for the min attribute

The specification states:

```
... Range state (type=range)
    ...
```

```
The step scale factor is 1. The default step is 1 (allowing only integers, unless the
min attribute has a non-integer value).
```

***EdgeHTML Mode***

The default step is incorrect if a non-integer value is specified for the `min` attribute.

## 2.1.43 [W3C-HTML52] Section 4.10.5.1.15. Checkbox state (type=checkbox)

V0161: The oninput event does not fire when the state of the checkbox is changed or when the click function is called

The specification states:

```
... Checkbox state (type=checkbox)
    ...
    Bookkeeping details
        ...
       ▪The input and change events apply.
```

***EdgeHTML Mode***

The `oninput` event does not fire when the state of the checkbox is changed or when the `click` function is called.

V0162: The checked state does not change when the check function is called

The specification states:

```
... Checkbox state (type=checkbox)
    ...
    If the element is mutable, then: The pre-click activation steps consist of setting
    the element's checkedness to its opposite value (i.e. true if it is false, false if
    it is true), and of setting the element's indeterminate IDL attribute to false. The
    canceled activation steps consist of setting the checkedness and the element's
    indeterminate IDL attribute back to the values they had before the pre-click
    activation steps were run. The activation behavior is to fire a simple event that
    bubbles named input at the element and then fire a simple event that bubbles named
    change at the element.
```

***EdgeHTML Mode***

The `checked` state does not change when the `check` function is called.

## 2.1.44 [W3C-HTML52] Section 4.10.5.1.16. Radio Button state (type=radio)

V0164: The oninput event does not fire if the state of the radio option is changed or the click function is called

The specification states:

```
... Radio Button state (type=radio)
    ...
    Bookkeeping details
        ...
        • The input and change events apply.
```

***EdgeHTML Mode***

The `oninput` event does not fire if the state of the radio option is changed or the `click` function is called.

V0166: The comparison of the name attributes is not done in a compatibility caseless manner for all Unicode ranges.

The specification states:

```
... Radio Button state (type=radio)
    ...
    The radio button group that contains an input element a also contains all the other
    input elements b that fulfill all of the following conditions:
        ...
        They both have a name attribute, their name attributes are not empty, and the
        value of a's name attribute is a compatibility caseless match for the value of
        b's name attribute.
```

***EdgeHTML Mode***

The comparison of the `name` attributes is not done in a compatibility caseless manner for all Unicode ranges; instead the comparison uses ASCII comparison.

V0168: When there are no checked elements, the checkedness values are set to false

The specification states:

```
... Radio Button state (type=radio)
    ...
    Constraint validation: If an element in the radio button group is required, and all
    of the input elements in the radio button group have a checkedness that is false,
    then the element is suffering from being missing.

        Note: If none of the radio buttons in a radio button group are checked when they
        are inserted into the document, then they will all be initially unchecked in the
        interface, until such time as one of them is checked (either by the user or by
        script).
```

***EdgeHTML Mode***

An `input type=radio` when part of a radio group that has no other checked elements within it, is considered, with all of the other radio group elements, to be in the intermediate state and all elements' `checkedness` values are set to `false`.

### 2.1.45 [W3C-HTML52] Section 4.10.5.1.17. File Upload state (type=file)

V0169: The file type does not properly secure the selected file

The specification states:

```
... File Upload state (type=file)
    ...
    For historical reasons, the value IDL attribute prefixes the file name with the
    string "C:\fakepath\". Some legacy user agents actually included the full path (which
    was a security vulnerability).
```

#### *EdgeHTML Mode*

The input type=`file` does not properly secure the selected file nor obscure the local file location. It obscures the file when it is submitted to the server.

### 2.1.46 [W3C-HTML52] Section 4.10.5.1.20. Reset Button state (type=reset)

V0170: Form controls linked using the form attribute are not reset

The specification states:

```
... Reset Button state (type=reset)
    ...
    If the element is mutable, then the element's activation behavior, if the element has
    a form owner and the element's Document is fully active, is to reset the form owner;
    otherwise, it is to do nothing.
```

#### *EdgeHTML Mode*

Form controls linked using the `form` attribute are not reset.

### 2.1.47 [W3C-HTML52] Section 4.10.7. The select element

V0175: The namedItem function throws an exception when it receives an empty string

The specification states:

```
4.10.7 The select element
    ...
    DOM interface:
        interface HTMLSelectElement : HTMLElement {
            ...
            HTMLOptionElement? namedItem(DOMString name);
            ...
        };
```

#### *EdgeHTML Mode*

The `namedItem` function throws an exception when it receives an empty string; it should return null.

### 2.1.48 [W3C-HTML52] Section 4.10.10. The option element

V0177: The text of nested SVG script elements is included in the returned value

The specification states:

```
4.10.10 The option element
    ...
    The text IDL attribute, on getting, must return the result of stripping and
    collapsing whitespace from the [child text content/concatenation of data of all the
    Text node descendants] of the option element, in tree order, excluding any that are
    descendants of descendants of the option element that are themselves script elements
    in the HTML namespace or script elements in the SVG namespace.
```

#### *EdgeHTML Mode*

The text of nested SVG script elements is included in the returned value.

### 2.1.49 [W3C-HTML52] Section 4.10.11. The textarea element

V0180: The autocomplete attribute is not supported

The specification states:

```
4.10.11 The textarea element
    ...
DOM interface:
        interface HTMLTextAreaElement : HTMLElement {
            attribute DOMString autocomplete;
            ...
        };
```

#### *EdgeHTML Mode*

The `autocomplete` attribute is not supported.

V0181: The dirName attribute is not supported

The specification states:

```
4.10.11 The textarea element
    ...
    DOM interface:
        interface HTMLTextAreaElement : HTMLElement {
            ...
            attribute DOMString dirName;
            ...
        };
```

#### *EdgeHTML Mode*

The `dirName` attribute is not supported.

### 2.1.50 [W3C-HTML52] Section 4.10.17.3. Association of controls and forms

V0190: The form attribute cannot be used to override the nearest ancestor form element

The specification states:

```
... Association of controls and forms
    ...
    A form-associated element is, by default, associated with its nearest ancestor form
    element (as described below), but, if it is reassociateable, may have a form
    attribute specified to override this.
```

#### *EdgeHTML Mode*

The `form` attribute cannot be used to override the nearest ancestor form element.

### 2.1.51 [W3C-HTML52] Section 4.10.18.2. Submitting element directionality: the dirname attribute

V0191: The dirname attribute is not supported

The specification states:

```
... Submitting element directionality: the dirname attribute

    The dirname attribute on a form control element enables the submission of the
    directionality of the element, and gives the name of the field that contains this
    value during form submission. If such an attribute is specified, its value must not
    be the empty string.
```

#### *EdgeHTML Mode*

The `dirname` attribute is not supported.

### 2.1.52 [W3C-HTML52] Section 4.10.18.7.1. Autofilling form controls: the autocomplete attribute

V0398: The autocomplete values on and off are not supported

The specification states:

```
4.10.18.7.1. Autofilling form controls: the autocomplete attribute
    ...
    When wearing the autofill expectation mantle, the autocomplete attribute, if
    specified, must have a value that is an ordered set of space-separated tokens
    consisting of either a single token that is an ASCII case-insensitive match for the
    string "off", or a single token that is an ASCII case-insensitive match for the
    string "on", or autofill detail tokens.

4.10.18.7.2 Processing model
    ...
    These values are defined as the result of running the following algorithm:
        ...
        7. If category is Off, let the element's autofill field name be the string "off",
           let its autofill hint set be empty, and let its IDL-exposed autofill value be
```

```
            the string "off". Then, abort these steps.

        8. If category is Automatic, let the element’s autofill field name be the string
           "on", let its autofill hint set be empty, and let its IDL-exposed autofill
           value be the string "on". Then, abort these steps.
```

***EdgeHTML Mode***

The autocomplete values on and off are not supported.

## 2.1.53 [W3C-HTML52] Section 4.10.18.7.2. Processing model

V0398: The autocomplete values on and off are not supported

The specification states:

```
4.10.18.7.1. Autofilling form controls: the autocomplete attribute
    ...
    When wearing the autofill expectation mantle, the autocomplete attribute, if
    specified, must have a value that is an ordered set of space-separated tokens
    consisting of either a single token that is an ASCII case-insensitive match for the
    string "off", or a single token that is an ASCII case-insensitive match for the
    string "on", or autofill detail tokens.

4.10.18.7.2 Processing model
    ...
    These values are defined as the result of running the following algorithm:
        ...
        7. If category is Off, let the element’s autofill field name be the string "off",
           let its autofill hint set be empty, and let its IDL-exposed autofill value be
           the string "off". Then, abort these steps.

        8. If category is Automatic, let the element’s autofill field name be the string
           "on", let its autofill hint set be empty, and let its IDL-exposed autofill
           value be the string "on". Then, abort these steps.
```

***EdgeHTML Mode***

The autocomplete values on and off are not supported.

## 2.1.54 [W3C-HTML52] Section 4.10.19. APIs for text field selections

V0195: The setRangeText functions are not supported

The specification states:

```
... APIs for the text field selections

    The input and textarea elements define the following members in their DOM interfaces
    for handling their selection: ... setRangeText(replacement) ...
```

***EdgeHTML Mode***

The `setRangeText` functions are not supported.

### 2.1.55 [W3C-HTML52] Section 4.10.20.2. Constraint validation

V0197: No list of elements is returned

The specification states:

```
... Constraint validation

    When the user agent is required to statically validate the constraints of form
    element form, it must run the following steps, which return either a positive result
    (all the controls in the form are valid) or a negative result (there are invalid
    controls) along with a (possibly empty) list of elements that are invalid and for
    which no script has claimed responsibility:
        ...
        7.  Return a negative result with the list of elements in the unhandled invalid
            controls list.
```

***EdgeHTML Mode***

No list of elements is returned.

### 2.1.56 [W3C-HTML52] Section 4.10.21.5. Selecting a form submission encoding

V0204: UTF-8 is used in form submission

The specification states:

```
... Selecting a form submission encoding

    If the user agent is to pick an encoding for a form ...
```

***EdgeHTML Mode***

UTF-8 is used in form submission even if accept-charset contains other encodings that can encode the entire form data set.

### 2.1.57 [W3C-HTML52] Section 4.10.21.6. URL-encoded form data

V0205: URL-encoded form data is encoded in UTF-8

The specification states:

```
... URL-encoded form data
    ...
    The application/x-www-form-urlencoded encoding algorithm is as follows:
```

***EdgeHTML Mode***

URL-encoded form data is encoded in UTF-8 regardless of what is in accept-charset.

V0206: URL-encoded form data includes the full filepath for type file, not the file name alone

The specification states:

```
... URL-encoded form data
    ...
    The application/x-www-form-urlencoded encoding algorithm is as follows:
```

***EdgeHTML Mode***

URL-encoded form data includes the full file path for type `file`, not the file name alone.

## 2.1.58 [W3C-HTML52] Section 4.10.21.7. Multipart form data

V0207: Forms are always encoded as UTF-8

The specification states:

```
... Multipart form data

    The multipart/form-data encoding algorithm is as follows:
        ...
        2.  If the algorithm was invoked with an explicit character encoding, let the
            selected character encoding be that encoding. ...
            ...
            Otherwise, let the selected character encoding be UTF-8.
```

***EdgeHTML Mode***

Forms are always encoded as UTF-8.

## 2.1.59 [W3C-HTML52] Section 4.10.21.8. Plain text form data

V0208: The submitted data set includes the full path of the file, not just the filename

The specification states:

```
... Plain text form data

    The text/plain encoding algorithm is as follows:
        ...
        5.  If the entry's type is "file", replace its value with the file's name only.
```

***EdgeHTML Mode***

The submitted data set includes the full path of the file, not just the filename.

## 2.1.60 [W3C-HTML52] Section 4.11.1. The details element

V0399: The details element is not supported

The specification states:

```
4.11.1 The details element
    ...
    The details element represents a disclosure widget from which the user can obtain
    additional information or controls.

10.5.3 The details element
    ...
    The details element is expected to render as a block box. ...
```

***EdgeHTML Mode***

The `details` element is not supported.

## 2.1.61 [W3C-HTML52] Section 4.11.2. The summary element

V0400: The summary element is not supported

The specification states:

```
4.11.2 The summary element
    ...
    The first summary child element of a details element represents a summary, caption,
    or legend for the rest of the contents of the parent details element, if any.
```

***EdgeHTML Mode***

The `summary` element is not supported.

## 2.1.62 [W3C-HTML52] Section 4.11.4. The dialog element

V0436: The dialog element is not supported

The specification states:

```
4.11.4. The dialog element
    …
    The dialog element represents a part of an application that a user interacts with to
    perform a task, for example a dialog box, inspector, or window.
```

***EdgeHTML Mode***

The `dialog` element is not supported.

## 2.1.63 [W3C-HTML52] Section 4.11.4.1. Anchor points

V0437: The anchor-point property is not supported

The specification states:

```
4.11.4.1. Anchor points

        This section will eventually be moved to a CSS specification; it is specified
        here only on an interim basis until an editor can be found to own this.

    …
    The anchor-point property specifies a point to which dialog boxes are to be aligned.
```

***EdgeHTML Mode***

The `anchor-point` property is not supported.

## 2.1.64 [W3C-HTML52] Section 4.12.1.1. Processing model

V0211: If the type attribute is an empty string value, it is not defaulted to text/javascript and JavaScript execution fails

The specification states:

```
... ...
    ...
    To prepare a script, the user agent must act as follows:
        ...
        6.  If either:

              o the script element has a type attribute and its value is the empty
              string, or

              o the script element has no type attribute but it has a language attribute
              and that attribute's value is the empty string, or

              o the script element has neither a type attribute nor a language attribute,
              then
            ... let the script block's type for this script element be "text/javascript".
```

***EdgeHTML Mode***

If the `type` attribute is an empty string value, it is not defaulted to `text/javascript` and JavaScript execution fails.

V0212: If the for attribute is not "window", or the event attribute is not "onload", the script is still executed

The specification states:

```
... ...
    ...
    To prepare a script, the user agent must act as follows:
        ...
        12. If the script element has an event attribute and a for attribute, ... then
            run these substeps:
            ...
            4.  If for is not an ASCII case-insensitive match for the string "window",
                then the user agent must abort these steps at this point. The script is
                not executed.

            5.  If event is not an ASCII case-insensitive match for either the string
```

```
"onload" or the string "onload()", then the user agent must abort these
steps at this point. The script is not executed.
```

***EdgeHTML Mode***

If the `for` attribute is not "window", or the `event` attribute is not "onload", the script is still executed.

## 2.1.65 [W3C-HTML52] Section 4.12.1.2. Scripting languages

V0215: Script language type recognition is based on the type/language type attribute

The specification states:

```
... Scripting languages
    ...
    When examining types to determine if they represent supported languages, user agents
    must not ignore MIME parameters. Types are to be compared including all parameters.
```

***EdgeHTML Mode***

Determination on whether a script is represented in a supported language is based on the type/language attribute. The attribute is validated against a list of recognized script types.

V0214: Some MIME types are not recognized

The specification states:

```
4.12.1.2 Scripting languages

    A JavaScript MIME type is a MIME type string that is one of the following and refers
    to JavaScript: [ECMA-262]

        "application/ecmascript"
        ...
        "text/javascript1.0"
        ...
        "text/javascript1.4"
        "text/javascript1.5"
        ...
        "text/x-ecmascript"

    User agents must recognize all JavaScript MIME types.
```

***EdgeHTML Mode***

The following MIME types are not recognized:

"application/x-ecmascript"

"text/javascript1.0"

"text/javascript1.4"

"text/javascript1.5"

"text/x-ecmascript"

### 2.1.66 [W3C-HTML52] Section 4.12.4. The canvas element

V0219: The width and height content attribute values truncate the content and do not return the value for getAttribute

The specification states:

```
... The canvas element
    ...
    Whenever the width and height content attributes are set, removed, changed, or
    redundantly set to the value they already have, if the canvas context mode is ... 2d,
    the user agent must set bitmap dimensions to the numeric values of the width and
    height content attributes.

    The width and height IDL attributes must reflect the respective content attributes of
    the same name, with the same defaults.
```

#### *EdgeHTML Mode*

The `width` and `height` content attribute values truncate the content and do not return the value for `getAttribute`.

### 2.1.67 [W3C-HTML52] Section 4.12.4.2. Serializing bitmaps to a file

V0220: Setting the JPEG quality to an invalid value does not cause the default value to be used

The specification states:

```
... Serializing bitmaps to a file
    ...
    Arguments for serialization methods [table]

        Other arguments [column]

            The second argument, if it is a number in the range 0.0 to 1.0 inclusive,
            must be treated as the desired quality level. If it is not a number or is
            outside that range, the user agent must use its default value, as if the
            argument had been omitted.
```

#### *EdgeHTML Mode*

Setting the JPEG quality to an invalid value does not cause the default value to be used.

### 2.1.68 [W3C-HTML52] Section 4.15.2. Pseudo-classes

V0221: The :active pseudo-class does not match appropriate ancestors of an element that matches

The specification states:

```
... Pseudo-classes
```

```
...
:active

    The :active pseudo-class is defined to match an element "while an element is
    being activated by the user".  ...
        ...
        ... element ... has a descendant that is currently matching the :active
        pseudo-class ...
```

***EdgeHTML Mode***

The :active pseudo-class does not match appropriate ancestors of an element that matches.

V0225: The :valid pseudo-class does not match a form element if the form owns a :valid candidate element

The specification states:

```
... Pseudo-classes
    ...
    :valid

        The :valid pseudo-class must match any element falling into one of the following
        categories:
            ...
            • form elements that are not the form owner of any elements that themselves
            are candidates for constraint validation but do not satisfy their constraints
```

***EdgeHTML Mode***

The :valid pseudo-class does not match a `form` element if the form owns a :valid candidate element.

V0226: The :valid pseudo-class does not match a fieldset element if that element has a child that is a :valid candidate element

The specification states:

```
... Pseudo-classes
    ...
    :valid

        The :valid pseudo-class must match any element falling into one of the following
        categories:
            ...
            • fieldset elements that have no descendant elements that themselves are
            candidates for constraint validation but do not satisfy their constraints
```

***EdgeHTML Mode***

The :valid pseudo-class does not match a `fieldset` element if that element has a child that is a :valid candidate element.

V0227: The :invalid pseudo-class does not match a form element if the form owns an :invalid candidate element

The specification states:

```
... Pseudo-classes
    ...
    :invalid

        The :invalid pseudo-class must match any element falling into one of the
        following categories:
            ...
            • form elements that are the form owner of one or more elements that
            themselves are candidates for constraint validation but do not satisfy their
            constraints
```

***EdgeHTML Mode***

The :invalid pseudo-class does not match a `form` element if the form owns an :invalid candidate element.

V0228: The :invalid pseudo-class does not match a fieldset element if that element has a child that is an :invalid candidate element

The specification states:

```
... Pseudo-classes
    ...
    :invalid

        The :invalid pseudo-class must match any element falling into one of the
        following categories:
            ..
            • fieldset elements that have of one or more descendant elements that
            themselves are candidates for constraint validation but do not satisfy their
            constraints
```

***EdgeHTML Mode***

The :invalid pseudo-class does not match a `fieldset` element if that element has a child that is an :invalid candidate element.

V0230: The :read-only and :read-write pseudo-classes are not supported

The specification states:

```
... Pseudo-classes
    ...
    :read-only
    :read-write

        The :read-write pseudo-class must match any element falling into one of the
        following categories, which for the purposes of Selectors are thus considered
        user-alterable:
            ...
        The :read-only pseudo-class must match all other HTML elements.
```

***EdgeHTML Mode***

The :read-only and :read-write pseudo-classes are not supported.

V0231: The :link pseudo-class does not match area or link elements with href attributes

The specification states:

```
... Pseudo-classes
    ...
    :link
    :visited

        All a elements that have an href attribute, all area elements that have an href
        attribute, and all link elements that have an href attribute, must match one of
        :link and :visited.
```

***EdgeHTML Mode***

The :link pseudo-class does not match `area` or `link` elements with `href` attributes.

V0232: The :indeterminate pseudo-class does not match an input[type=radio] element when no option is selected

The specification states:

```
... Pseudo-classes
    ...
    :indeterminate

        The :indeterminate pseudo-class must match any element falling into one of the
        following categories:
            ...
            • input elements whose type attribute is in the Radio Button state and whose
            radio button group contains no input elements whose checkedness state is true.
```

***EdgeHTML Mode***

The :indeterminate pseudo-class does not match the an `input` element whose type attribute is in the Radio Button state and whose radio button group contains no input elements whose checkedness state is true.

V0233: The :valid and :invalid pseudo-classes can match an element even if the constraints are violated by the initial value defined on the element

The specification states:

```
... Pseudo-classes
    ...
    :valid

        The :valid pseudo-class must match any element falling into one of the following
        categories:
            ...

    :invalid
```

```
The :invalid pseudo-class must match any element falling into one of the
following categories:
    ...
```

***EdgeHTML Mode***

The :valid and :invalid pseudo-classes can match an element even if the constraints are violated by the initial value defined on the element.

### 2.1.69 [W3C-HTML52] Section 5.3. Activation

V0019: A synthetic click does not set the isTrusted flag

The specification states:

```
... ...
    ...
    When a user agent is to run synthetic click activation steps on an element, the user
    agent must run the following steps:
        ...
        4. Fire a click event at the element. If the run synthetic click activation steps
           algorithm was invoked because the click() method was invoked, then the
           isTrusted attribute must be initialized to false.
```

***EdgeHTML Mode***

A synthetic click does not set the `isTrusted` flag.

### 2.1.70 [W3C-HTML52] Section 5.4.3. The tabindex attribute

V0297: tabindex returns 0 as the default for elements that are not focusable

The specification states:

```
... ...
    ...
    The tabIndex IDL attribute must reflect the value of the tabindex content attribute.
    Its default value is 0 for elements that are focusable and –1 for elements that are
    not focusable.
```

***EdgeHTML Mode***

`tabindex` returns `0` as the default for elements that are not focusable.

### 2.1.71 [W3C-HTML52] Section 5.6.1. Making document regions editable: The contenteditable content attribute

V0298: The contentEditable attribute does not return inherit when its value is set to the empty string

The specification states:

```
... Making document regions editable: The contenteditable content attribute
    ...
    The contenteditable ... attribute is an enumerated attribute whose keywords are the
    empty string, true, and false. The empty string and the true keyword map to the true
    state. The false keyword maps to the false state. In addition, there is a third
    state, the inherit state, which is the missing value default (and the invalid value
    default).
```

***EdgeHTML Mode***

The `contentEditable` attribute does not return `inherit` when its value is set to the empty string.

V0299: An invalid value for the contentEditable attribute throws an "Invalid Argument" exception.

The specification states:

```
... Making document regions editable: The contenteditable content attribute
    ...
    The contentEditable IDL attribute, on getting, must return the string "true" if the
    content attribute is set to the true state, "false" if the content attribute is set
    to the false state, and "inherit" otherwise. On setting, if the new value is an ASCII
    case-insensitive match for the string "inherit" then the content attribute must be
    removed, if the new value is an ASCII case-insensitive match for the string "true"
    then the content attribute must be set to the string "true", if the new value is an
    ASCII case-insensitive match for the string "false" then the content attribute must
    be set to the string "false", and otherwise the attribute setter must throw a
    ["SyntaxError" DOMException/SyntaxError exception].
```

***EdgeHTML Mode***

An invalid value for the `contentEditable` attribute throws an "Invalid Argument" exception.

## 2.1.72 [W3C-HTML52] Section 5.6.5. Spelling and grammar checking

V0301: The spellcheck attribute cannot be set to override the default when the default is true and the element was created using createElement

The specification states:

```
... Spelling and grammar checking
    ...
    element . spellcheck [ = value ]

        Returns true if the element is to have its spelling and grammar checked;
        otherwise, returns false.

        Can be set, to override the default and set the spellcheck content attribute.
```

***EdgeHTML Mode***

The `spellcheck` attribute cannot be set to override the default when the default is `true` and the element was created using `createElement`.

### 2.1.73 [W3C-HTML52] Section 5.7.3.1. The DataTransferItemList interface

V0406: The getter is not properly supported

The specification states:

```
5.7.3.1. The DataTransferItemList interface

    Each DataTransfer object is associated with a DataTransferItemList object.

        interface DataTransferItemList {
            ...
            getter DataTransferItem (unsigned long index);
            ...
        };
```

***EdgeHTML Mode***

The getter is not properly supported and is defined as `item`.

```
getter File item(unsigned long index);
```

### 2.1.74 [W3C-HTML52] Section 5.7.4. The DragEvent interface

V0407: There is no constructor defined for the DragEvent interface

The specification states:

```
5.7.4. The DragEvent interface

    The drag-and-drop processing model involves several events. They all use the
    DragEvent interface.

        [Constructor(DOMString type, optional DragEventInit eventInitDict)]
        interface DragEvent : MouseEvent {
             readonly attribute DataTransfer? dataTransfer;
        };
```

***EdgeHTML Mode***

There is no constructor defined for the `DragEvent` interface.

V0408: The dataTransfer attribute is not defined as nullable

The specification states:

```
5.7.4. The DragEvent interface

    The drag-and-drop processing model involves several events. They all use the
    DragEvent interface.

        [Constructor(DOMString type, optional DragEventInit eventInitDict)]
        interface DragEvent : MouseEvent {
            readonly attribute DataTransfer? dataTransfer;
        };
```

***EdgeHTML Mode***

The `dataTransfer` attribute is not defined as nullable.

```
readonly attribute DataTransfer dataTransfer;
```

### 2.1.75 [W3C-HTML52] Section 6.1. Browsing contexts

V0410: The first Document is for the home page specified in the preferences, not for about:blank.

The specification states:

```
6.1 Browsing contexts

    A browsing context is an environment in which Document objects are presented to the
    user.
    ...
    To create a new browsing context:
        ...
        3.  Let document be a new Document, whose URL is about:blank, which is marked as
            being an HTML document, whose character encoding is UTF-8, and which is both
            ready for post-load tasks and completely loaded immediately.
```

***EdgeHTML Mode***

The first `Document` is for the home page specified in the preferences, not for `about:blank`.

V0411: A newly created browsing context does not change the readyState from loading to complete

The specification states:

```
6.1 Browsing contexts

    A browsing context is an environment in which Document objects are presented to the
    user.
    ...
    To create a new browsing context:
        ...
        3.  Let document be a new Document, whose URL is about:blank, which is marked as
            being an HTML document, whose character encoding is UTF-8, and which is both
            ready for post-load tasks and completely loaded immediately.
```

***EdgeHTML Mode***

A newly created browsing context does not change the `readyState` from `loading` to `complete`.

V0412: The referrer is set to the empty string, not the creator URL

The specification states:

```
6.1 Browsing contexts

    A browsing context is an environment in which Document objects are presented to the
    user.
    ...
```

```
To create a new browsing context:
    ...
    5.  If the new browsing context has a creator browsing context, then set
        document's referrer to the creator URL.
```

***EdgeHTML Mode***

The referrer is set to the empty string, not the creator URL.

V0238: The about:blank document does not have a character encoding of UTF-8

The specification states:

```
6.1 Browsing contexts
    ...
    To create a new browsing context:
        ...
        3.  Let document be a new Document, whose URL is about:blank, which is marked as
            being an HTML document, whose character encoding is UTF-8, and which is both
            ready for post-load tasks and completely loaded immediately.
```

***EdgeHTML Mode***

The `about:blank` document does not have a character encoding of UTF-8.

## 2.1.76 [W3C-HTML52] Section 6.1.1.1. Navigating nested browsing contexts in the DOM

V0413: In a nested browsing context, the frameElement attribute returns undefined, not an object

The specification states:

```
6.1.1.1 Navigating nested browsing contexts in the DOM
    ...
    The frameElement IDL attribute, on getting, must run the following algorithm:
        1.  Let d be the Window object's associated Document.
        2.  Let context be d's browsing context.
        3.  If context is not a nested browsing context, return null and abort these
            steps.
        4.  Let container be context's browsing context container.
        5.  If container's node document's origin is not same origin-domain with the
            entry settings object's origin, then return null and abort these steps.
        6.  Return container.
```

***EdgeHTML Mode***

In a nested browsing context, the `frameElement` attribute returns undefined, not an object.

## 2.1.77 [W3C-HTML52] Section 6.1.5. Browsing context names

V0241: An empty string for a context name prevents navigation within an iframe and attempts to show a popup

The specification states:

```
... Browsing context names
    ...
    The rules for choosing a browsing context given a browsing context name are as
    follows. The rules assume that they are being applied in the context of a browsing
    context, as part of the execution of a task.

        1. If the given browsing context name is the empty string or _self, then the
           chosen browsing context must be the current one.
```

***EdgeHTML Mode***

An empty string for a context name prevents navigation within an iframe and attempts to show a popup.

V0242: Links with the noreferrer keyword do not create a new browsing context

The specification states:

```
... Browsing context names
    ...
    The rules for choosing a browsing context given a browsing context name are as
    follows. The rules assume that they are being applied in the context of a browsing
    context, as part of the execution of a task.
        ...
        5.  Otherwise, a new browsing context is being requested, and what happens
            depends on the user agent's configuration and abilities — it is determined by
            the rules given for the first applicable option from the following list:
            ...
            If the user agent has been configured such that in this instance it will
            create a new browsing context, and the browsing context is being requested as
            part of following a hyperlink whose link types include the noreferrer keyword

                A new top-level browsing context must be created. If the given browsing
                context name is not _blank, then the new top-level browsing context's
                name must be the given browsing context name (otherwise, it has no name).
                The chosen browsing context must be this new browsing context. The
                creation of such a browsing context is a new start for session storage.
```

***EdgeHTML Mode***

Links with the `noreferrer` keyword do not create a new browsing context.

## 2.1.78 [W3C-HTML52] Section 6.3. The Window object

V0415: Some attributes and methods do not return a WindowProxy

The specification states:

```
6.3. The Window object

    [PrimaryGlobal, LegacyUnenumerableNamedProperties]
    /*sealed*/ interface Window : EventTarget {
        // the current browsing context
        [Unforgeable] readonly attribute WindowProxy window;
        [Replaceable] readonly attribute WindowProxy self;
```

```
        ...
        // other browsing contexts
        [Replaceable] readonly attribute WindowProxy frames;
        ...
        [Unforgeable] readonly attribute WindowProxy top;
        ...
        [Replaceable] readonly attribute WindowProxy parent;
        ...
        WindowProxy open(optional DOMString url = "about:blank", optional DOMString
        target = "_blank",
          [TreatNullAs=EmptyString] optional DOMString features = "", optional boolean
          replace = false);
        getter WindowProxy (unsigned long index);
        ...
    };
```

***EdgeHTML Mode***

Some attributes and methods do not return a WindowProxy; instead they return a Window.

[ Unforgeable ] readonly attribute Window window;

[ Replaceable ] readonly attribute Window self;

[ Replaceable ] readonly attribute Window frames;

[ Unforgeable ] readonly attribute Window top;

[ Replaceable ] readonly attribute Window parent;

Window open(optional DOMString url = "about:blank", optional DOMString target = "_blank",

[TreatNullAs=EmptyString] optional DOMString features = "", optional boolean replace = false);

getter Window (unsigned long index);

V0416: The frameElement attribute is not nullable

The specification states:

```
6.3. The Window object

    [PrimaryGlobal, LegacyUnenumerableNamedProperties]
    interface Window : EventTarget {
        ...
        readonly attribute Element? frameElement;
        ...
    };
```

***EdgeHTML Mode***

The `frameElement` attribute is not nullable.

readonly attribute Element frameElement;

### 2.1.79 [W3C-HTML52] Section 6.3.1. APIs for creating and navigating browsing contexts by name

V0249: No InvalidAccessError exception is thrown when the target argument does not result in a valid browsing context name

The specification states:

```
... APIs for creating and navigating browsing contexts by name
    ...
    If ... [this results/would result] in there not being a chosen browsing context, then
    throw an InvalidAccessError exception and abort these steps.
```

#### *EdgeHTML Mode*

No `InvalidAccessError` exception is thrown when the target argument does not result in a valid browsing context name.

### 2.1.80 [W3C-HTML52] Section 6.3.2. Accessing other browsing contexts

V0251: The length attribute does not return the number of child browsing contexts

The specification states:

```
5.2.3 Accessing other browsing contexts
    ...
    The length IDL attribute['s getter] ... must return the number of child browsing
    contexts ... .
```

#### *EdgeHTML Mode*

The `length` attribute does not return the number of child browsing contexts of the active document.

### 2.1.81 [W3C-HTML52] Section 6.3.3. Named access on the Window object

V0253: Framesets are not identifiable by name

The specification states:

```
... Named access on the Window object
    ...
    Named objects with the name name, for the purposes of the above algorithm, are those
    that are either:
        ...
        a, applet, area, embed, form, frameset, img, or object elements that have a name
        content attribute whose value is name, or
        ...
```

#### *EdgeHTML Mode*

Framesets are not identifiable by name.

### 2.1.82 [W3C-HTML52] Section 6.4.1. Relaxing the same-origin restriction

V0417: No SecurityError exceptiont is thrown if there is no browsing context et al.

The specification states:

```
6.4.1. Relaxing the same-origin restriction
    ...
    The domain attribute on setting must run these steps:
        1.  If this Document object has no browsing context, throw a "SecurityError"
            DOMException.
        2.  If this Document object's active sandboxing flag set has its sandboxed
            document.domain
            browsing context flag set, then throw a "SecurityError" DOMException.
        3.  If the given value is the empty string, then throw a "SecurityError"
            DOMException.
        4.  Let host be the result of parsing the given value.
        5.  If host is failure, then throw a "SecurityError" DOMException.
        6.  Let effectiveDomain be this Document object's origin's effective domain.
        7.  If host is not equal to effectiveDomain, then run these substeps:
            1.  If host or effectiveDomain is not domain, then throw a "SecurityError"
                DOMException.
                NOTE:
                This is meant to exclude hosts that are an IPv4 address or an IPv6
                address.
            2.  If host, prefixed by a U+002E FULL STOP (.), does not exactly match the
                effectiveDomain,
                then throw a "SecurityError" DOMException.
            3.  If host matches a suffix in the Public Suffix List, or, if host, prefixed
                by a U+002E FULL STOP (.),
                matches the end of a suffix in the Public Suffix List, then throw a
                "SecurityError" DOMException. [PSL]
                Suffixes must be compared after applying the host parser algorithm. [URL]
        8.  Set origin's domain to host.
```

#### *EdgeHTML Mode*

No `SecurityError` exception is thrown if: there is no browsing context, the sandbox flag is set, and the new value is not exactly equal to the current value of `document.domain`.

### 2.1.83 [W3C-HTML52] Section 6.6.1. The session history of browsing contexts

V0236: Nested browsing contexts share a session history

The specification states:

```
... The session history of browsing contexts
    ...
    The sequence of Documents in a browsing context is its session history. Each browsing
    context, including nested browsing contexts, has a distinct session history. ...
```

#### *EdgeHTML Mode*

Nested browsing contexts share a session history.

### 2.1.84 [W3C-HTML52] Section 6.7.6. Page load processing model for media

V0256: Audio and video media are not loaded into a Document; instead a download is attempted

The specification states:

```
... Page load processing model for media

    When an image, video, or audio resource is to be loaded in a browsing context, the
    user agent should create a Document object, mark it as being an HTML document, set
    its content type to the ... MIME type of the resource (type in the navigate
    algorithm), ... append an html element to the Document, append a head element and a
    body element to the html element, append an element host element for the media, as
    described below, to the body element, and set the appropriate attribute of the
    element host element, as described below, to the address of the image, video, or
    audio resource.
```

***EdgeHTML Mode***

Audio and video media are not loaded into a `Document`; instead a download is attempted.

### 2.1.85 [W3C-HTML52] Section 6.7.7. Page load processing model for content that uses plugins

V0257: Plugins are not loaded into a newly created document; instead a download is attempted

The specification states:

```
... Page load processing model for content that uses plugins

    When a resource that requires an external resource to be rendered is to be loaded in
    a browsing context, the user agent should create a Document object, mark it as being
    an HTML document and mark it as being a plugin document, set its content type to the
    ... MIME type of the resource (type in the navigate algorithm), ... append an html
    element to the Document, append a head element and a body element to the html
    element, append an embed to the body element, and set the src attribute of the embed
    element to the address of the resource.
```

***EdgeHTML Mode***

Plugins are not loaded into a newly created document; instead a download is attempted.

### 2.1.86 [W3C-HTML52] Section 6.7.11. Unloading documents

V0262: The ignore-opens-during-unload counter is not modified

The specification states:

```
... Unloading documents
    ...
    When a user agent is to prompt to unload a document, it must run the following steps.
        ...
        2.  Increase the Document's ignore-opens-during-unload counter by one.
```

***EdgeHTML Mode***

The `ignore-opens-during-unload` counter is not modified.

## 2.1.87 [W3C-HTML52] Section 7.1.3.10.2. The PromiseRejectionEvent interface

V0433: The PromiseRejectionEvent is not supported

The specification states:

```
7.1.3.10.2. The PromiseRejectionEvent interface

    [Constructor(DOMString type, PromiseRejectionEventInit eventInitDict),
    Exposed=(Window,Worker)]
    interface PromiseRejectionEvent : Event {
        readonly attribute Promise<any> promise;
        readonly attribute any reason;
    };
```

***EdgeHTML Mode***

The PromiseRejectionEvent is not supported.

## 2.1.88 [W3C-HTML52] Section 7.1.5.1. Event handlers

V0268: The body.onload event does not overwrite the window.onload event

The specification states:

```
... Event handlers
    ...
    If an event handler IDL attribute exposes an event handler of an object that doesn't
    exist, it must always return null on getting and must do nothing on setting.
```

***EdgeHTML Mode***

The `body.onload` event does not overwrite the `window.onload` event.

V0271: The form owner is not taken into account in events

The specification states:

```
... Event handlers
    ...
    When the user agent is to get the current value of the event handler H, it must run
    these steps:
        ...
        1.  If H's value is an internal raw uncompiled handler, run these substeps:
            ...
            5.  If element is not null and element has a form owner, let form owner be
                that form owner. Otherwise, let form owner be null.
```

***EdgeHTML Mode***

The form owner is not taken into account in events.

### 2.1.89 [W3C-HTML52] Section 7.1.5.2. Event handlers on elements, Documentobjects, and Window objects

V0272: The oncancel event handler is not supported

The specification states:

```
... Event handlers on elements, Document objects, and Window objects

    The following are the event handlers (and their corresponding event handler event
    types) that must be supported by all HTML elements, as both event handler content
    attributes and event handler IDL attributes; and that must be supported by all
    Document and Window objects, as event handler IDL attributes:
        ...
        oncancel    cancel
```

***EdgeHTML Mode***

The `oncancel` event handler is not supported.

V0273: The oncuechange event handler is not supported on Document or Window

The specification states:

```
... Event handlers on elements, Document objects, and Window objects
    ...
    The following are the event handlers (and their corresponding event handler event
    types) that must be supported by all HTML elements, as both event handler content
    attributes and event handler IDL attributes; and that must be supported by all
    Document and Window objects, as event handler IDL attributes:
        ...
        oncuechange    cuechange
```

***EdgeHTML Mode***

The `oncuechange` event handler is not supported on `Document` or `Window`.

V0276: The onresize event handler is not supported on Document or HTML elements

The specification states:

```
... Event handlers on elements, Document objects, and Window objects
    ...
    The following are the event handlers (and their corresponding event handler event
    types) that must be supported by all HTML elements other than body and frameset
    elements, as both event handler content attributes and event handler IDL attributes;
    that must be supported by all Document objects, as event handler IDL attributes; and
    that must be supported by all Window objects, as event handler IDL attributes on the
    Window objects themselves, and with corresponding event handler content attributes
    and event handler IDL attributes exposed on all body and frameset elements that are
```

```
owned by that Window object's Documents:
    ...
    onresize    resize
```

***EdgeHTML Mode***

The `onresize` event handler is not supported on `Document` or `HTML` elements.

V0277: The onshow event handler is not supported

The specification states:

```
... Event handlers on elements, Document objects, and Window objects
    ...
    The following are the event handlers (and their corresponding event handler event
    types) that must be supported by all HTML elements, as both event handler content
    attributes and event handler IDL attributes; and that must be supported by all
    Document and Window objects, as event handler IDL attributes:
        ...
        onshow    show
```

***EdgeHTML Mode***

The `onshow` event handler is not supported.

V0278: The ontoggle event handler is not supported

The specification states:

```
... Event handlers on elements, Document objects, and Window objects
    ...
    The following are the event handlers (and their corresponding event handler event
    types) that must be supported by all HTML elements, as both event handler content
    attributes and event handler IDL attributes; and that must be supported by all
    Document and Window objects, as event handler IDL attributes:
        ...
        ontoggle    toggle
```

***EdgeHTML Mode***

The `ontoggle` event handler is not supported.

V0279: The onpopstate event handler is not supported on the frameset element

The specification states:

```
... Event handlers on elements, Document objects, and Window objects
    ...
    The following are the event handlers (and their corresponding event handler event
    types) that must be supported by Window objects, as event handler IDL attributes on
    the Window objects themselves, and with corresponding event handler content
    attributes and event handler IDL attributes exposed on all body and frameset elements
    that are owned by that Window object's Documents:
        ...
```

```
onpopstate    popstate
```

***EdgeHTML Mode***

The `onpopstate` event handler is not supported on the `frameset` element.

V0373: The onmouseenter and onmouseleave event handlers are not supported on Document

The specification states:

```
... Event handlers on elements, Document objects, and Window objects
    ...
    The following are the event handlers (and their corresponding event handler event
    types) that must be supported by all HTML elements, as both event handler content
    attributes and event handler IDL attributes; and that must be supported by all
    Document and Window objects, as event handler IDL attributes:
        ...
        onmouseenter    mouseenter
        onmouseleave    mouseleave
```

***EdgeHTML Mode***

The `onmouseenter` and `onmouseleave` event handlers are not supported on `Document`.

## 2.1.90 [W3C-HTML52] Section 7.2. The WindowOrWorkerGlobalScope mixin

V0434: The createImageBitmap methods are not supported

The specification states:

```
7.2. The WindowOrWorkerGlobalScope mixin


    The WindowOrWorkerGlobalScope mixin is for use of APIs that are to be exposed on
    Window and WorkerGlobalScope objects.

        [NoInterfaceObject, Exposed=(Window, Worker)]
        interface WindowOrWorkerGlobalScope {
            …
            // ImageBitmap, Images (ImageBitmapFactories)
            Promise<ImageBitmap> createImageBitmap(ImageBitmapSource image);
            Promise<ImageBitmap> createImageBitmap(ImageBitmapSource image, long sx, long
            sy, long sw, long sh);
        };
```

***EdgeHTML Mode***

The createImageBitmap methods are not supported.

## 2.1.91 [W3C-HTML52] Section 7.4.1. Opening the input stream

V0280: The document object is not reused after window.open is called

The specification states:

```
... Opening the input stream

    The open() method comes in several variants with different numbers of arguments.

        document = document . open( [ type [, replace ] ] )

            Causes the Document to be replaced in-place, as if it was a new Document
            object, but reusing the previous object, which is then returned.
```

***EdgeHTML Mode***

The document object is not reused after `window.open` is called.

V0281: The salvageable state of the Document is not set when the Document is unloaded

The specification states:

```
... Opening the input stream
    ...
    When called with two arguments ..., the document.open() method must act as follows:
        ...
        8.  Set the Document's salvageable state to false.
```

***EdgeHTML Mode***

The salvageable state of the `Document` is not set when the `Document` is unloaded.

V0283: The script-created parser is not freed from the script stack

The specification states:

```
... Opening the input stream
    ...
    When called with two arguments ... , the document.open() method must act as follows:
        ...
        ... Create a new HTML parser and associate it with the document. This is a
        script-created parser (meaning that it can be closed by the document.open() and
        document.close() methods, and that the tokenizer will wait for an explicit call
        to document.close() before emitting an end-of-file token). The encoding
        confidence is irrelevant.
```

***EdgeHTML Mode***

The script-created parser is not freed from the script stack.

V0282: Singleton objects are not replaced for location, history, navigator, applicationCache, sessionStorage, or localStorage

The specification states:

```
7.4.1. Opening the input stream
    ...
```

```
When called with two arguments (or fewer), the document.open() method must act as
follows:
    ...
    18. Replace the Document's singleton objects with new instances of those objects,
        created in window's Realm. (This includes in particular the History and
        Navigator objects, the various BarProp objects, the two Storage objects, the
        various HTMLCollection objects, and objects defined by other specifications,
        like Selection. It also includes all the Web IDL prototypes in the JavaScript
        binding, including the Document object's prototype.)
```

***EdgeHTML Mode***

Singleton objects are not replaced for `location`, `history`, `navigator`, `applicationCache`, `sessionStorage`, or `localStorage`.

## 2.1.92 [W3C-HTML52] Section 7.4.2. Closing the input stream

V0284: No InvalidStateError exception is thrown for a Document object not flagged as an HTML document

The specification states:

```
... Closing the input stream
    ...
    The close() method must run the following steps:

        1.  If the Document object is [an XML document/not flagged as an HTML document],
            throw an InvalidStateError exception and abort these steps.
```

***EdgeHTML Mode***

No `InvalidStateError` exception is thrown for a Document object not flagged as an HTML document.

## 2.1.93 [W3C-HTML52] Section 7.4.3. document.write()

V0285: No InvalidStateError exception is thrown for a Document object not flagged as an HTML document

The specification states:

```
... document.write()
    ...
    The document.write(...) method must act as follows:

        1. If the method was invoked on an XML document, throw an InvalidStateError
           exception and abort these steps.
```

***EdgeHTML Mode***

No `InvalidStateError` exception is thrown for a Document object not flagged as an HTML document.

### 2.1.94 [W3C-HTML52] Section 7.7.1.3. Custom scheme and content handlers: the registerProtocolHandler() and registerContentHandler() methods

V0432: Custom scheme and content handlers are not supported

The specification states:

```
7.7.1.3. Custom scheme and content handlers: the registerProtocolHandler() and
registerContentHandler() methods

        interface NavigatorContentUtils {
            // content handler registration
            void registerProtocolHandler(DOMString scheme, DOMString url, DOMString
            title);
            void registerContentHandler(DOMString mimeType, DOMString url, DOMString
            title);
            DOMString isProtocolHandlerRegistered(DOMString scheme, DOMString url);
            DOMString isContentHandlerRegistered(DOMString mimeType, DOMString url);
            void unregisterProtocolHandler(DOMString scheme, DOMString url);
            void unregisterContentHandler(DOMString mimeType, DOMString url);
        };

The registerProtocolHandler() method allows Web sites to register themselves as possible
handlers for particular schemes.
```

***EdgeHTML Mode***

Custom scheme and content handlers are not supported.

### 2.1.95 [W3C-HTML52] Section 7.8. Images

V0424: The ImageBitmap interface is not supported

The specification states:

```
... Images

        [Exposed=(Window, Worker)]
        interface ImageBitmap {
            ...
        };

An ImageBitmap object represents a bitmap image that can be painted to a canvas without
undue latency.
```

***EdgeHTML Mode***

The `ImageBitmap` interface is not supported.

### 2.1.96 [W3C-HTML52] Section 8.2. Parsing HTML documents

V0302: Listed parse errors do not properly change states to the data state when an error occurs

The specification states:

```
8.2 Parsing HTML documents
```

```
...
This specification defines the parsing rules for HTML documents, whether they are
syntactically correct or not. Certain points in the parsing algorithm are said to be
parse errors. The error handling for parse errors is well-defined (that's the
processing rules described throughout this specification), but user agents, while
parsing an HTML document, may abort the parser at the first parse error that they
encounter for which they do not wish to apply the rules described in this
specification.
```

***EdgeHTML Mode***

Listed parse errors do not properly change states to the data state when an error occurs.

## 2.1.97 [W3C-HTML52] Section 8.2.3.1. The insertion mode

V0304: There is no check of last, and no switch to the "in head" insertion mode

The specification states:

```
8.2.3.1 The insertion mode

    The insertion mode is a state variable that controls the primary operation of the
    tree construction stage.
    ...
    When the steps below require the UA to reset the insertion mode appropriately, it
    means the user agent must follow these steps:
        ...
        ...  If node is a head element and last is false, then switch the insertion mode
        to "in head" and abort these steps.
```

***EdgeHTML Mode***

There is no check of `last`, and no switch to the "in head" insertion mode.

## 2.1.98 [W3C-HTML52] Section 8.2.3.2. The stack of open elements

V0305: Non-HTML namespace nested elements do not close table elements of the HTML namespace

The specification states:

```
8.2.3.2 The stack of open elements
    ...
    The stack of open elements is said to have a particular element in table scope when
    it has that element in the specific scope consisting of the following element types:

        ▪ html in the HTML namespace
        ▪ table in the HTML namespace
        ▪ template in the HTML namespace
```

***EdgeHTML Mode***

Non-HTML namespace nested elements do not close `table` elements of the HTML namespace.

### 2.1.99 [W3C-HTML52] Section 8.2.4.36. Attribute value (double-quoted) state

V0306: NULL character U+0000 does not produce a parse error prior to changing the character to the replacement character U+FFFD

The specification states:

```
... Attribute value (double-quoted) state

    Consume the next input character:
        ...
        U+0000 NULL
            Parse error. Append a U+FFFD REPLACEMENT CHARACTER character to the current
            attribute's value.
```

***EdgeHTML Mode***

NULL character U+0000 does not produce a parse error prior to changing the character to the replacement character U+FFFD.

### 2.1.100 [W3C-HTML52] Section 8.2.4.37. Attribute value (single-quoted) state

V0307: NULL character U+0000 does not produce a parse error prior to changing the character to the replacement character U+FFFD

The specification states:

```
... Attribute value (single-quoted) state

    Consume the next input character:
        ...
        U+0000 NULL

            Parse error. Append a U+FFFD REPLACEMENT CHARACTER character to the current
            attribute's value.
```

***EdgeHTML Mode***

NULL character U+0000 does not produce a parse error prior to changing the character to the replacement character U+FFFD.

### 2.1.101 [W3C-HTML52] Section 8.2.4.42. Markup declaration open state

V0308: A non-HTML namespace CDATA section is not consumed properly and does not switch state

The specification states:

```
... Markup declaration open state
    ...
    Otherwise, if there is an adjusted current node and it is not an element in the HTML
    namespace and the next seven characters are a case-sensitive match for the string
    "[CDATA[" (the five uppercase letters "CDATA" with a U+005B LEFT SQUARE BRACKET
    character before and after), then consume those characters and switch to the CDATA
    section state.
```

***EdgeHTML Mode***

A non-HTML namespace CDATA section is not consumed properly and does not switch state.

### 2.1.102 [W3C-HTML52] Section 8.2.5. Tree construction

V0310: MathML is not supported

The specification states:

```
8.2.5 Tree construction
    ...
    A node is a MathML text integration point if it is one of the following elements:
```

***EdgeHTML Mode***

MathML is not supported.

### 2.1.103 [W3C-HTML52] Section 8.2.5.3. Closing elements that have implied end tags

V0311: The rb and rtc elements are not supported and do not cause implied end tags to be generated

The specification states:

```
8.2.5.3 Closing elements that have implied end tags

    When the steps below require the UA to generate implied end tags, then, while the
    current node is a dd element, a dt element, an li element, an optgroup element, an
    option element, a p element, an rb element, an rp element, an rt element, or an rtc
    element, the UA must pop the current node off the stack of open elements.
```

***EdgeHTML Mode***

The `rb` and `rtc` elements are not supported and do not cause implied end tags to be generated.

### 2.1.104 [W3C-HTML52] Section 8.2.5.4.7. The "in body" insertion mode

V0312: node is not removed from the list of active formatting elements

The specification states:

```
8.2.5.4.7 The "in body" insertion mode
    ...
    The adoption agency algorithm, which takes as its only argument ... for which the
    algorithm is being run, consists of the following steps:
        ...
        ...  Let node and last node be furthest block. Follow these steps:
            ...
            5.  If inner loop counter is greater than three and node is in the list of
                active formatting elements, then remove node from the list of active
```

```
formatting elements.
```

***EdgeHTML Mode***

*node* is not removed from the list of active formatting elements.

## 2.1.105 [W3C-HTML52] Section 8.2.5.4.9. The "in table" insertion mode

V0313: An input element within a table does not acknowledge the token's self-closing flag for the input element

The specification states:

```
8.2.5.4.9 The "in table" insertion mode

    When the user agent is to apply the rules for the "in table" insertion mode, the user
    agent must handle the token as follows:
        ...
        A start tag whose tag name is "input"

            If the token does not have an attribute with the name "type", or if it does,
            but that attribute's value is not an ASCII case-insensitive match for the
            string "hidden", then: act as described in the "anything else" entry below.

            Otherwise:

            Parse error.

            Insert an HTML element for the token.

            Pop that input element off the stack of open elements.

            Acknowledge the token's self-closing flag, if it is set.
```

***EdgeHTML Mode***

An `input` element within a table does not acknowledge the token's self-closing flag for the `input` element.

## 2.1.106 [W3C-HTML52] Section 8.2.5.4.11. The "in caption" insertion mode

V0314: Tags in a caption tag do not properly pop elements off the stack, clear the active formatting elements, or switch to "in table" insertion mode

The specification states:

```
8.2.5.4.11 The "in caption" insertion mode

    When the user agent is to apply the rules for the "in caption" insertion mode, the
    user agent must handle the token as follows:
        ...
        A start tag whose tag name is one of: "caption", "col", "colgroup", "tbody",
        "td", "tfoot", "th", "thead", "tr"

        An end tag whose tag name is "table"
            ...
```

```
Otherwise
...
Pop elements from this stack until a caption element has been popped from the
stack.

Clear the list of active formatting elements up to the last marker.

Switch the insertion mode to "in table".

Reprocess the token.
```

***EdgeHTML Mode***

The start tags `caption`, `col`, `colgroup`, `tbody`, `td`, `tfoot`, `th`, `thead`, `tr` and the end tag `table` when nested within an open `caption` tag do not properly pop elements off the stack, clear the active formatting elements, or switch to "in table" insertion mode.

### 2.1.107 [W3C-HTML52] Section 8.2.5.4.17. The "in select in table" insertion mode

V0315: End tags within a select tag within a table are not processed correctly and are ignored

The specification states:

```
8.2.5.4.17 The "in select in table" insertion mode

    When the user agent is to apply the rules for the "in select in table" insertion
    mode, the user agent must handle the token as follows:
        ...
        An end tag whose tag name is one of: "caption", "table", "tbody", "tfoot",
        "thead", "tr", "td", "th"

            Parse error.

            If the stack of open elements does not have an element in table scope that is
            an HTML element and with the same tag name as that of the token, then ignore
            the token.

            Otherwise:

            Pop elements from the stack of open elements until a select element has been
            popped from the stack.

            Reset the insertion mode appropriately.

            Reprocess the token.
```

***EdgeHTML Mode***

End tags `caption`, `table`, `tbody`, `tfoot`, `thead`, `tr`, `td` and `th` within a `select` tag within a table are not processed correctly and are ignored.

### 2.1.108 [W3C-HTML52] Section 8.2.5.5. The rules for parsing tokens in foreign content

V0317: A U+0000 NULL character does not generate a parse error

The specification states:

```
8.2.5.5 The rules for parsing tokens in foreign content

    When the user agent is to apply the rules for parsing tokens in foreign content, the
    user agent must handle the token as follows:

        A character token that is U+0000 NULL
            Parse error. Insert a U+FFFD REPLACEMENT CHARACTER character.
```

***EdgeHTML Mode***

A U+0000 NULL character does not generate a parse error.

## 2.1.109 [W3C-HTML52] Section 10.3.1. Hidden elements

V0318: The area, base, basefont, link, param, rp, source, template, and track elements do not set a default style of display: none

The specification states:

```
10.3.1 Hidden elements
    ...
    [hidden], area, base, basefont, datalist, head, link, ... meta,
    noembed, noframes, param, rp, script, source, style, template, track, title {
        display: none;
    }
```

***EdgeHTML Mode***

The `area`, `base`, `basefont`, `link`, `param`, `rp`, `source`, `template`, and `track` elements do not set a default style of `display: none`.

V0319: The default style of the noframes element is set to display: block, not display: none

The specification states:

```
10.3.1 Hidden elements
    ...
    [hidden], area, base, basefont, datalist, head, link, ... meta,
    noembed, noframes, param, rp, script, source, style, template, track, title {
        display: none;
    }
```

***EdgeHTML Mode***

The default style of the `noframes` element is set to `display: block`, not `display: none`.

V0320: The embed element when hidden does not set the default styles

The specification states:

```
10.3.1 Hidden elements
    ...
    embed[hidden] { display: inline; height: 0; width: 0; }
```

***EdgeHTML Mode***

The `embed` element when hidden does not set the default styles. (It only hides the element.)

## 2.1.110 [W3C-HTML52] Section 10.3.3. Flow content

V0321: The legend element is not set to display: block

The specification states:

```
10.3.3 Flow content
    ...
    address, blockquote, center, div, figure, figcaption, footer, form, header, hr,
    legend, listing, ... p, plaintext, pre, ..., xmp {
        display: block;
    }
```

***EdgeHTML Mode***

The `legend` element is not set to `display: block`.

V0323: The listing, plaintext, pre, and xmp elements do not set the font-family property to monospace

The specification states:

```
10.3.3 Flow content
    ...
    listing, plaintext, pre, xmp {
        font-family: monospace; white-space: pre;
    }
```

***EdgeHTML Mode***

The `listing`, `plaintext`, `pre`, and `xmp` elements do not set the `font-family` property to `monospace`.

V0324: The pre element, when the wrap attribute is specified, does not set the white-space property to pre-wrap

The specification states:

```
10.3.3 Flow content
    ...
    pre[wrap] { white-space: pre-wrap; }
```

***EdgeHTML Mode***

The `pre` element, when the `wrap` attribute is specified, does not set the `white-space` property to `pre-wrap`.

### 2.1.111 [W3C-HTML52] Section 10.3.4. Phrasing content

V0325: The b and strong elements set font-weight: bold instead of font-weight: bolder

The specification states:

```
10.3.4 Phrasing content
    ...
    b, strong { font-weight: bolder; }
```

#### *EdgeHTML Mode*

The `b` and `strong` elements set `font-weight: bold` instead of `font-weight: bolder`.

V0326: The big element does not set font-size:larger in the default styles

The specification states:

```
10.3.4 Phrasing content
    ...
    big { font-size: larger; }
```

#### *EdgeHTML Mode*

The `big` element does not set `font-size: larger` in the default styles.

V0327: The elements small, sub, and sup do not set font-size: smaller in the default styles

The specification states:

```
10.3.4 Phrasing content
    ...
    small { font-size: smaller; }
    ...
    sub, sup { line-height: normal; font-size: smaller; }
```

#### *EdgeHTML Mode*

The elements `small`, `sub`, and `sup` do not set `font-size: smaller` in the default styles.

V0328: The elements sub and sup do not set line-height in the default styles

The specification states:

```
10.3.4 Phrasing content
    ...
```

```
sub, sup { line-height: normal; font-size: smaller; }
```

***EdgeHTML Mode***

The elements `sub` and `sup` do not set `line-height` in the default styles.

V0331: Styles are not set for the :link and :visited states in the default styles

The specification states:

```
10.3.4 Phrasing content
    ...
    :link { color: #0000EE; }
    :visited { color: #551A8B; }
    ...
    :link, :visited { text-decoration: underline; ... }
    a:link[rel~=help], a:visited[rel~=help],
    area:link[rel~=help], area:visited[rel~=help] { cursor: help; }
```

***EdgeHTML Mode***

The following styles are not set for the `:link` and `:visited` states in the default styles:

:link { color: #0000EE; }

:visited { color: #551A8B; }

:link, :visited { text-decoration: underline; ... }

a:link[rel~=help], a:visited[rel~=help],

area:link[rel~=help], area:visited[rel~=help] { cursor: help; }

V0332: The abbr and acronym elements do not set text-decoration: dotted underline default styles

The specification states:

```
10.3.4 Phrasing content
    ...
    abbr[title], acronym[title] { text-decoration: dotted underline; }
```

***EdgeHTML Mode***

The `abbr` and `acronym` elements do not set `text-decoration: dotted underline` in default styles.

V0333: The blink element does not have a defined default style

The specification states:

```
10.3.4 Phrasing content
    ...
    blink { text-decoration: blink; }
```

***EdgeHTML Mode***

The `blink` element does not have a defined default style.

V0335: The size attribute of the font element sets the font-size property to the wrong value

The specification states:

```
10.3.4 Phrasing content
    ...
    When a font element has a size attribute, the user agent is expected to use the
    following steps, known as the rules for parsing a legacy font size, to treat the
    attribute as a presentational hint setting the element's 'font-size' property:
        ...
        12. Set 'font-size' to the keyword corresponding to the value of value according
            to the following table:

                value | 'font-size' keyword | notes
                ---------------------------------
                1     | x-small             |
                2     | small               |
                3     | medium              |
                4     | large               |
                5     | x-large             |
                6     | xx-large            |
                7     | xxx-large           | see below

            The 'xxx-large' value is a non-CSS value used here to indicate a font size
            50% larger than 'xx-large'.
```

***EdgeHTML Mode***

The size attribute of the font element sets the font-size property to the wrong value:

```
value | 'font-size' keyword | notes

---------------------------------

1     | xx-small            |

2     | x-small             |

3     | small               |

4     | medium              |

5     | large               |

6     | x-large             |

7     | xx-large            |
```

V0334: The br, nobr, and wbr elements do not set any default styles

The specification states:

```
10.3.4 Phrasing content
    ...
    br { display-outside: newline; } /* this also has bidi implications */
    nobr { white-space: nowrap; }
```

```
wbr { display-outside: break-opportunity; } /* this also has bidi implications */
nobr wbr { white-space: normal; }
...
br[clear=left i] { clear: left; }
br[clear=right i] { clear: right; }
br[clear=all i], br[clear=both i] { clear: both; }
```

***EdgeHTML Mode***

The `br`, `nobr`, and `wbr` elements do not set any default styles.

### 2.1.112 [W3C-HTML52] Section 10.3.5. Bidirectional text

V0426: All bidirectional text default styles are set incorrectly

The specification states:

```
10.3.5 Bidirectional text
    ...
    [dir]:dir(ltr), bdi:dir(ltr), input[type=tel i]:dir(ltr) { direction: ltr; }
    [dir]:dir(rtl), bdi:dir(rtl) { direction: rtl; }

    address, blockquote, center, div, figure, figcaption, footer, form, header, hr,
    legend, listing, main, p, plaintext, pre, summary, xmp, article, aside, h1, h2,
    h3, h4, h5, h6, ... nav, section, table, caption, colgroup, col, thead,
    tbody, tfoot, tr, td, th, dir, dd, dl, dt, ... ol, ul, li, bdi, output,
    [dir=ltr i], [dir=rtl i], [dir=auto i] {
        unicode-bidi: isolate;
    }

    bdo, bdo[dir] { unicode-bidi: isolate-override; }

    input[dir=auto i]:matches([type=search i], [type=tel i], [type=url i],
    [type=email i]), textarea[dir=auto i], pre[dir=auto i] {
        unicode-bidi: plaintext;
    }
```

***EdgeHTML Mode***

All bidirectional text default styles are set incorrectly. The CSS `:dir` selector is not supported and `unicode-bidi` values `isolate`, `isolate-override`, and `plaintext` are not supported.

### 2.1.113 [W3C-HTML52] Section 10.3.6. Quotes

V0337: No quote values are defined in the default styles

The specification states:

```
10.3.6 Quotes
    ...
    User agents are expected to use either the block below (which will be regularly
    updated) or to automatically generate their own copy directly from the source
    material. ...
```

***EdgeHTML Mode***

No quote values are defined in the default styles.

### 2.1.114 [W3C-HTML52] Section 10.3.7. Sections and headings

V0427: No nesting rules for sections and headings are defined

The specification states:

```
10.3.7 Sections and headings
    ...
    article, aside, h1, h2, h3, h4, h5, h6, nav, section {
        display: block;
    }

    h1 { margin-top: 0.67em; margin-bottom: 0.67em; font-size: 2.00em; font-weight: bold;
    }
    h2 { margin-top: 0.83em; margin-bottom: 0.83em; font-size: 1.50em; font-weight: bold;
    }
    h3 { margin-top: 1.00em; margin-bottom: 1.00em; font-size: 1.17em; font-weight: bold;
    }
    h4 { margin-top: 1.33em; margin-bottom: 1.33em; font-size: 1.00em; font-weight: bold;
    }
    h5 { margin-top: 1.67em; margin-bottom: 1.67em; font-size: 0.83em; font-weight: bold;
    }
    h6 { margin-top: 2.33em; margin-bottom: 2.33em; font-size: 0.67em; font-weight: bold;
    }

    In the following CSS block, x is shorthand for the following selector:
    :matches(article, aside, nav, section)
    ...
    x h1 { margin-top: 0.83em; margin-bottom: 0.83em; font-size: 1.50em; }
    x x h1 { margin-top: 1.00em; margin-bottom: 1.00em; font-size: 1.17em; }
    x x x h1 { margin-top: 1.33em; margin-bottom: 1.33em; font-size: 1.00em; }
    x x x x h1 { margin-top: 1.67em; margin-bottom: 1.67em; font-size: 0.83em; }
    x x x x x h1 { margin-top: 2.33em; margin-bottom: 2.33em; font-size: 0.67em; }
```

#### *EdgeHTML Mode*

No nesting rules for sections and headings are defined.

### 2.1.115 [W3C-HTML52] Section 10.3.8. Lists

V0341: The elements ol and li do support the default styles for an attribute value of type=A, upper-alpha

The specification states:

```
10.3.8 Lists
    ...
    ol[type=A], li[type=A] { list-style-type: upper-alpha; }
```

#### *EdgeHTML Mode*

The elements `ol` and `li` do not support the default styles for an attribute value of `type=A`, `upper-alpha`.

V0428: The dl element does not set margins within the default styles

The specification states:

```
10.3.8 Lists
    ...
    dir, dl, ol, ul { margin-block-start: 1em; margin-block-end: 1em; }

    :matches(dir, dl, ol, ul) :matches(dir, dl, ol, ul) {
        margin-block-start: 0; margin-block-end: 0;
    }
```

***EdgeHTML Mode***

The `dl` element does not set margins within the default styles.

V0339: The dd element does not properly account for direction for default style margin settings

The specification states:

```
10.3.8 Lists
    ...
    dd { margin-inline-start: 40px; }
```

***EdgeHTML Mode***

The `dd` element does not properly account for direction for default style margin settings.

## 2.1.116 [W3C-HTML52] Section 10.3.9. Tables

V0342: The table element does not set the text-indent: initial default style

The specification states:

```
10.3.9 Tables
    ...
    table {
        box-sizing: border-box;
        border-spacing: 2px;
        border-collapse: separate;
        text-indent: initial;
    }
```

***EdgeHTML Mode***

The `table` element does not set the `text-indent: initial` default style.

V0343: The table, td, and th elements do not set the correct border colors in the default styles

The specification states:

```
10.3.9 Tables
```

```
...
table, td, th { border-color: gray; }
thead, tbody, tfoot, tr { border-color: inherit; }
table[rules=none i], table[rules=groups i], table[rules=rows i],
table[rules=cols i], table[rules=all i], table[frame=void i],
table[frame=above i], table[frame=below i], table[frame=hsides i],
table[frame=lhs i], table[frame=rhs i], table[frame=vsides i],
table[frame=box i], table[frame=border i],
table[rules=none i] > tr > td, table[rules=none i] > tr > th,
table[rules=groups i] > tr > td, table[rules=groups i] > tr > th,
table[rules=rows i] > tr > td, table[rules=rows i] > tr > th,
table[rules=cols i] > tr > td, table[rules=cols i] > tr > th,
table[rules=all i] > tr > td, table[rules=all i] > tr > th,
table[rules=none i] > thead > tr > td, table[rules=none i] > thead > tr > th,
table[rules=groups i] > thead > tr > td, table[rules=groups i] > thead > tr > th,
table[rules=rows i] > thead > tr > td, table[rules=rows i] > thead > tr > th,
table[rules=cols i] > thead > tr > td, table[rules=cols i] > thead > tr > th,
table[rules=all i] > thead > tr > td, table[rules=all i] > thead > tr > th,
table[rules=none i] > tbody > tr > td, table[rules=none i] > tbody > tr > th,
table[rules=groups i] > tbody > tr > td, table[rules=groups i] > tbody > tr > th,
table[rules=rows i] > tbody > tr > td, table[rules=rows i] > tbody > tr > th,
table[rules=cols i] > tbody > tr > td, table[rules=cols i] > tbody > tr > th,
table[rules=all i] > tbody > tr > td, table[rules=all i] > tbody > tr > th,
table[rules=none i] > tfoot > tr > td, table[rules=none i] > tfoot > tr > th,
table[rules=groups i] > tfoot > tr > td, table[rules=groups i] > tfoot > tr > th,
table[rules=rows i] > tfoot > tr > td, table[rules=rows i] > tfoot > tr > th,
table[rules=cols i] > tfoot > tr > td, table[rules=cols i] > tfoot > tr > th,
table[rules=all i] > tfoot > tr > td, table[rules=all i] > tfoot > tr > th {
    border-color: black;
}
```

***EdgeHTML Mode***

The `table`, `td`, and `th` elements do not set the border colors in the default styles. Any border colors that are set are defaulted to gray.

V0429: Background images on table elements are aligned based on the table element, not relative to their respectively applied element

The specification states:

```
10.3.9 Tables
    ...
    When a table, thead, tbody, tfoot, tr, td, or th element has a background attribute
    set to a non-empty value, the new value is expected to be parsed relative to the
    element’s node document, and if this is successful, the user agent is expected to
    treat the attribute as a presentational hint setting the element’s background-image
    property to the resulting URL string.
```

***EdgeHTML Mode***

Background images on `table` elements are aligned based on the `table` element, not relative to their respectively applied element.

V0345: The default styles for the table element's frame and rules attributes are not properly defined

The specification states:

10.3.9 Tables

```
    ...
    table[rules=none i], table[rules=groups i], table[rules=rows i],
    table[rules=cols i], table[rules=all i] {
        border-style: hidden;
        border-collapse: collapse;
    }
    table[border] { border-style: outset; } /* only if border is not equivalent to zero */
    table[frame=void i] { border-style: hidden; }
    table[frame=above i] { border-style: outset hidden hidden hidden; }
    table[frame=below i] { border-style: hidden hidden outset hidden; }
    table[frame=hsides i] { border-style: outset hidden outset hidden; }
    table[frame=lhs i] { border-style: hidden hidden hidden outset; }
    table[frame=rhs i] { border-style: hidden outset hidden hidden; }
    table[frame=vsides i] { border-style: hidden outset; }
    table[frame=box i], table[frame=border i] { border-style: outset; }

    table[border] > tr > td, table[border] > tr > th,
    table[border] > thead > tr > td, table[border] > thead > tr > th,
    table[border] > tbody > tr > td, table[border] > tbody > tr > th,
    table[border] > tfoot > tr > td, table[border] > tfoot > tr > th {
        /* only if border is not equivalent to zero */
        border-width: 1px;
        border-style: inset;
    }
    table[rules=none i] > tr > td, table[rules=none i] > tr > th,
    table[rules=none i] > thead > tr > td, table[rules=none i] > thead > tr > th,
    table[rules=none i] > tbody > tr > td, table[rules=none i] > tbody > tr > th,
    table[rules=none i] > tfoot > tr > td, table[rules=none i] > tfoot > tr > th,
    table[rules=groups i] > tr > td, table[rules=groups i] > tr > th,
    table[rules=groups i] > thead > tr > td, table[rules=groups i] > thead > tr > th,
    table[rules=groups i] > tbody > tr > td, table[rules=groups i] > tbody > tr > th,
    table[rules=groups i] > tfoot > tr > td, table[rules=groups i] > tfoot > tr > th,
    table[rules=rows i] > tr > td, table[rules=rows i] > tr > th,
    table[rules=rows i] > thead > tr > td, table[rules=rows i] > thead > tr > th,
    table[rules=rows i] > tbody > tr > td, table[rules=rows i] > tbody > tr > th,
    table[rules=rows i] > tfoot > tr > td, table[rules=rows i] > tfoot > tr > th {
        border-width: 1px;
        border-style: none;
    }
    table[rules=cols i] > tr > td, table[rules=cols i] > tr > th,
    table[rules=cols i] > thead > tr > td, table[rules=cols i] > thead > tr > th,
    table[rules=cols i] > tbody > tr > td, table[rules=cols i] > tbody > tr > th,
    table[rules=cols i] > tfoot > tr > td, table[rules=cols i] > tfoot > tr > th {
        border-width: 1px;
        block-start-style: none;
        border-inline-end-style: solid;
        border-block-end-style: none;
        border-inline-start-style: solid;
    }
    table[rules=all i] > tr > td, table[rules=all i] > tr > th,
    table[rules=all i] > thead > tr > td, table[rules=all i] > thead > tr > th,
    table[rules=all i] > tbody > tr > td, table[rules=all i] > tbody > tr > th,
    table[rules=all i] > tfoot > tr > td, table[rules=all i] > tfoot > tr > th {
        border-width: 1px;
        border-style: solid;
    }

    table[rules=groups i] > colgroup {
        border-inline-start-width: 1px;
        border-inline-start-style: solid;
        border-inline-end-width: 1px;
        border-inline-end-style: solid;
    }
    table[rules=groups i] > thead,
    table[rules=groups i] > tbody,
    table[rules=groups i] > tfoot {
        border-block-start-width: 1px;
        border-block-start-style: solid;
```

```
            border-block-end-width: 1px;
            border-block-end-style: solid;
        }

    table[rules=rows i] > tr, table[rules=rows i] > thead > tr,
    table[rules=rows i] > tbody > tr, table[rules=rows i] > tfoot > tr {
      border-block-start-width: 1px;
      border-block-start-style: solid;
      border-block-end-width: 1px;
      border-block-end-style: solid;
    }
```

***EdgeHTML Mode***

The default styles for the `table` element's `frame` and `rules` attributes are not properly defined because the case-insensitive matching within the CSS attribute selector is not supported.

V0353: When the align attribute is set, the default styles do not set the margin properties

The specification states:

```
    10.3.9 Tables
        ...
        table[align=center i] { margin-inline-start: auto; margin-inline-end: auto; }

    10.3.12 The hr element
        ...
        hr[align=left i] { margin-left: 0; margin-right: auto; }
        hr[align=right i] { margin-left: auto; margin-right: 0; }
        hr[align=center i] { margin-left: auto; margin-right: auto; }
```

***EdgeHTML Mode***

When the `align` attribute is set, the default styles do not set the `margin` properties.

## 2.1.117 [W3C-HTML52] Section 10.3.11. Form controls

V0349: The input, select, option, optgroup, button, textarea and keygen elements do not set text-indent: initial in default styles

The specification states:

```
    10.3.11 Form controls
        ...
        input, select, option, optgroup, button, textarea, keygen {
            text-indent: initial;
        }
```

***EdgeHTML Mode***

The `input`, `select`, `option`, `optgroup`, `button`, `textarea`, and `keygen` elements do not set `text-indent: initial` in default styles.

V0430: All input controls are set to box-sizing: border-box

The specification states:

```
10.3.11 Form controls
    ...
    input:matches([type=radio i], [type=checkbox i], [type=reset i], [type=button i],
    [type=submit i], [type=search i]), select, button {
        box-sizing: border-box;
    }
```

***EdgeHTML Mode***

All input controls, not just the `radio`, `checkbox`, `reset`, `button`, and `submit` controls, are set to `box-sizing: border-box`.

## 2.1.118 [W3C-HTML52] Section 10.3.12. The hr element

V0353: When the align attribute is set, the default styles do not set the margin properties

The specification states:

```
10.3.9 Tables
    ...
    table[align=center i] { margin-inline-start: auto; margin-inline-end: auto; }

10.3.12 The hr element
    ...
    hr[align=left i] { margin-left: 0; margin-right: auto; }
    hr[align=right i] { margin-left: auto; margin-right: 0; }
    hr[align=center i] { margin-left: auto; margin-right: auto; }
```

***EdgeHTML Mode***

When the `align` attribute is set, the default styles do not set the `margin` properties.

V0352: The color property for the hr element is not set to gray in the default styles

The specification states:

```
10.3.12 The hr element
    ...
    hr {
        color: gray;
        border-style: inset;
        border-width: 1px;
        margin-block-start: 0.5em;
        margin-inline-end: auto;
        margin-block-end: 0.5em;
        margin-inline-start: auto;
    }
```

***EdgeHTML Mode***

The `color` property for the `hr` element is not set to `gray` in the default styles. Instead the color used is `rgb(0, 0, 0)`, which is equivalent to `black`.

### 2.1.119 [W3C-HTML52] Section 10.3.13. The fieldset and legend elements

V0354: The fieldset element does not set the padding values or the border styles correctly in the default styles

The specification states:

```
10.3.13 The fieldset and legend elements
    …
    fieldset {
        display: block;
        margin-inline-start: 2px;
        margin-inline-end: 2px;
        border: groove 2px ThreeDFace;
        padding-block-start: 0.35em;
        padding-inline-end: 0.625em;
        padding-block-end: 0.75em;
        padding-inline-start: 0.625em;
        min-width: min-content;
    }
```

***EdgeHTML Mode***

The `fieldset` element does not set the padding values or the border styles correctly in the default styles; instead it uses the value `groove 2px gray`.

### 2.1.120 [W3C-HTML52] Section 10.4.1. Embedded content

V0355: No default styles are applied to the video element

The specification states:

```
10.4.1 Embedded content
    ...
    The following CSS rules are expected to apply:
        ...
        video { object-fit: contain; }
```

***EdgeHTML Mode***

No default styles are applied to the `video` element.

### 2.1.121 [W3C-HTML52] Section 10.4.2. Images

V0357: When the image does not load, the input element of type=image does not render as a button

The specification states:

```
10.4.2 Images
```

```
...
User agents are expected to render img elements and input elements whose type
attributes are in the Image Button state, according to the first applicable rules
from the following list:
    ...
    If the element is an input element that does not represent an image and the user
    agent does not expect this to change

        The user agent is expected to treat the element as a replaced element
        consisting of a button whose content is the element's alternative text. The
        intrinsic dimensions of the button are expected to be about one line in
        height and whatever width is necessary to render the text on one line.
```

***EdgeHTML Mode***

When the image does not load, the `input` element of `type=image` does not render as a button.

## 2.1.122 [W3C-HTML52] Section 10.4.3. Attributes for embedded content and images

V0358: Default styles are not defined for align attributes on replaced elements

The specification states:

```
10.4.3 Attributes for embedded content and images
    ...
    iframe[frameborder=0], iframe[frameborder=no i] { border: none; }

    applet[align=left i], embed[align=left i], iframe[align=left i],
    img[align=left i], input[type=image i][align=left i], object[align=left i] {
        float: left;
    }

    applet[align=right i], embed[align=right i], iframe[align=right i],
    img[align=right i], input[type=image i][align=right i], object[align=right i] {
        float: right;
    }

    applet[align=top i], embed[align=top i], iframe[align=top i],
    img[align=top i], input[type=image i][align=top i], object[align=top i] {
        vertical-align: top;
    }

    applet[align=baseline i], embed[align=baseline i], iframe[align=baseline i],
    img[align=baseline i], input[type=image i][align=baseline i], object[align=baseline
    i] {
        vertical-align: baseline;
    }

    applet[align=texttop i], embed[align=texttop i], iframe[align=texttop i],
    img[align=texttop i], input[type=image i][align=texttop i], object[align=texttop i] {
        vertical-align: text-top;
    }

    applet[align=absmiddle i], embed[align=absmiddle i], iframe[align=absmiddle i],
    img[align=absmiddle i], input[type=image i][align=absmiddle i],
    object[align=absmiddle i],
    applet[align=abscenter i], embed[align=abscenter i], iframe[align=abscenter i],
    img[align=abscenter i], input[type=image i][align=abscenter i],
    object[align=abscenter i] {
        vertical-align: middle;
    }
```

```
applet[align=bottom i], embed[align=bottom i], iframe[align=bottom i],
img[align=bottom i], input[type=image i][align=bottom i],
object[align=bottom i] {
    vertical-align: bottom;
}
```

***EdgeHTML Mode***

Default styles are not defined for `align` attributes on replaced elements.

### 2.1.123 [W3C-HTML52] Section 10.4.4. Image maps

V0359: A CSS cursor value set on the area element does not override settings on the img or object elements

The specification states:

```
10.4.4 Image maps

    Shapes on an image map are expected to act, for the purpose of the CSS cascade, as
    elements independent of the original area element that happen to match the same style
    rules but inherit from the img or object element.

    For the purposes of the rendering, only the 'cursor' property is expected to have any
    effect on the shape.
```

***EdgeHTML Mode***

A CSS `cursor` value set on the `area` element does not override settings on the `img` or `object` elements.

### 2.1.124 [W3C-HTML52] Section 10.5.3. The details and summary elements

V0399: The details element is not supported

The specification states:

```
4.11.1 The details element
    ...
    The details element represents a disclosure widget from which the user can obtain
    additional information or controls.

10.5.3 The details element
    ...
    The details element is expected to render as a block box. ...
```

***EdgeHTML Mode***

The `details` element is not supported.

### 2.1.125 [W3C-HTML52] Section 11.3.1. The applet element

V0431: The applet element is no longer supported

The specification states:

```
11.3.1. The applet element
This feature is in the process of being removed from the Web platform. (This is a long
process that takes many years.) Using the applet element at this time is highly
discouraged.
```

#### *EdgeHTML Mode*

The applet element is no longer supported and has been removed from the platform.

### 2.1.126 [W3C-HTML52] Section 11.3.4. Other elements, attributes and APIs

V0365: The noHref attribute of the area element incorrectly returns -1 when set to true

The specification states:

```
... Other elements, attributes and APIs
    ...
    The noHref IDL attribute of the area element must reflect the element's nohref
    content attribute.
```

#### *EdgeHTML Mode*

The `noHref` attribute of the `area` element incorrectly returns -1 when set to `true`.

V0368: The align attribute of the input element does not return the value specified

The specification states:

```
... Other elements, attributes and APIs
    ...
    The align IDL attribute of the input element must reflect the content attribute of
    the same name.
```

#### *EdgeHTML Mode*

The `align` attribute of the `input` element does not return the value specified.

V0366: The align attribute of the embed element is not supported

The specification states:

```
... Other elements, attributes and APIs
    ...
    The name and align IDL attributes of the embed element must reflect the respective
    content attributes of the same name.
```

***EdgeHTML Mode***

The `align` attribute of the `embed` element is not supported.

### 2.1.127 [W3C-HTML52] Section 11.3.4.1. Plugins

V0295: The NavigatorPlugins interface is not supported

The specification states:

```
... Plugins

    [NoInterfaceObject]
    interface NavigatorPlugins {
        ...
    };
```

***EdgeHTML Mode***

The `NavigatorPlugins` interface is not supported.

## 2.2 Clarifications

There are no clarifications of the MAY and SHOULD requirements of [W3C-HTML52].

### 2.2.1 [W3C-HTML52] Section 2.2.1. Conformance classes

C0001: The developer tools preserve the conformance errors and indicate errors either with a message in a console window or with a red underline for the specific error

The specification states:

```
2.2.1 Conformance classes
    ...
    Authoring tools and markup generators
        ...
        When an authoring tool is used to edit a non-conforming document, it may preserve
        the conformance errors in sections of the document that were not edited during
        the editing session (i.e. an editing tool is allowed to round-trip erroneous
        content). However, an authoring tool must not claim that the output is conformant
        if errors have been so preserved.
```

***EdgeHTML Mode***

The developer tools preserve the conformance errors and indicate errors either with a message in a console window or with a red underline for the specific error.

C0002: Many platform restrictions are in place to prevent denial of service attacks

The specification states:

```
2.2.1 Conformance classes
```

```
…
Authoring tools and markup generators
...
    User agents may impose implementation-specific limits on otherwise unconstrained
    inputs, e.g. to prevent denial of service attacks, to guard against running out
    of memory, or to work around platform-specific limitations.
```

***EdgeHTML Mode***

Many platform restrictions are in place to prevent denial of service attacks.

### 2.2.2 [W3C-HTML52] Section 2.2.2. Dependencies

C0003: The WebVTT specification is a supported text track format for media resources

The specification states:

```
2.2.2 Dependencies
    ...
    WebVTT
        Implementations may support WebVTT as a text track format for subtitles,
        captions, chapter titles, metadata, etc, for media resources. [WEBVTT]
```

***EdgeHTML Mode***

The WebVTT specification is a supported text track format for media resources.

### 2.2.3 [W3C-HTML52] Section 2.6.3. Encrypted HTTP and related security concerns

C0004: There is no warning if the user visits a page that uses less secure encryption than it did on a prior visit by that user

The specification states:

```
2.6.3 Encrypted HTTP and related security concerns
    ...
    User agents should warn the user that there is a potential problem whenever the user
    visits a page that the user has previously visited, if the page uses less secure
    encryption on the second visit.
```

***EdgeHTML Mode***

There is no warning if the user visits a page that uses less secure encryption than it did on a prior visit by that user.

### 2.2.4 [W3C-HTML52] Section 3.2.5.1. The title attribute

C0006: There is no indicator for elements that have a title attribute set

The specification states:

```
... The title attribute
    ...
    User agents should inform the user when elements have advisory information, otherwise
    the information would not be discoverable.
```

***EdgeHTML Mode***

There is no indicator for elements that have a `title` attribute set.

### 2.2.5 [W3C-HTML52] Section 3.2.5.2. The lang and xml:lang attributes

C0007: The lang attribute is used to determine which fonts and quotes to use

The specification states:

```
... The lang and xml:lang attributes
    ...
    User agents may use the element's language to determine proper processing or
    rendering (e.g. in the selection of appropriate fonts or pronunciations, for
    dictionary selection, or for the user interfaces of form controls such as date
    pickers).
```

***EdgeHTML Mode***

The `lang` attribute is used to determine which fonts and quotes to use within a document.

### 2.2.6 [W3C-HTML52] Section 4.2.4.2. Processing the type attribute

C0010: When necessary the image sniffing rules are used to determine the official type

The specification states:

```
4.2.4.2. Processing the type attribute
    ...
    ... Otherwise, if the resource is expected to be an image, user agents may apply the
    image sniffing rules, with the official type being the type determined from the
    resource's Content-Type metadata, and use the resulting ... type of the resource as
    if it was the actual type. ...
```

***EdgeHTML Mode***

When necessary the image sniffing rules are used to determine the official type.

### 2.2.7 [W3C-HTML52] Section 4.2.4.3. Obtaining a resource from a link element

C0009: Resources are obtained as needed unless a prefetch flag is set

The specification states:

```
4.2.4.3. Obtaining a resource from a link element
    ...
```

```
User agents may opt to only try to obtain such resources when they are needed,
instead of pro-actively fetching all the external resources that are not applied.
```

***EdgeHTML Mode***

Resources are obtained as needed. Proactive fetching occurs only when a specific `prefetch` flag is set.

### 2.2.8 [W3C-HTML52] Section 4.2.4.5. Providing users with a means to follow hyperlinks created using the link element

C0008: There is no direct way for the user to access the hyperlinks created by the link element

The specification states:

```
4.2.4.5. Providing users with a means to follow hyperlinks created using the link element
    ...
    Interactive user agents may provide users with a means to follow the hyperlinks
    created using the link element, somewhere within their user interface. ...
```

***EdgeHTML Mode***

There is no direct way for the user to access the hyperlinks created by the `link` element. However, there is programmatic access to the information through the `link` element itself.

### 2.2.9 [W3C-HTML52] Section 4.2.5.1. Standard metadata names

C0085: The text "This site says..." is used for UI in cases of page-created dialogs

The specification states:

```
4.2.5.1 Standard metadata names
    This specification defines a few names for the name attribute of the meta element.
    ...
    application-name
        ...
        User agents may use the application name in UI in preference to the page's title,
        since the title might include status messages and the like relevant to the status
        of the page at a particular moment in time instead of just being the name of the
        application.
```

***EdgeHTML Mode***

The text "This site says..." is used for UI in cases of page-created dialogs.

### 2.2.10 [W3C-HTML52] Section 4.2.5.3. Pragma directives

C0012: There is no visual representation of timers or redirects, but there are indicators for link destinations

The specification states:

```
4.2.5.3 Pragma directives
    ...
    Refresh state (http-equiv="refresh")

        This pragma acts as timed redirect.
            ...
            ... Perform one or more of the following steps:
                    ...
                In addition, the user agent may, as with anything, inform the user of any
                and all aspects of its operation, including the state of any timers, the
                destinations of any timed redirects, and so forth.
```

***EdgeHTML Mode***

There is no visual representation of timers or redirects. However, there are indicators for destinations when hovering over a link.

## 2.2.11 [W3C-HTML52] Section 4.4.5. The blockquote element

C0014: There is no way for the user to follow citation links

The specification states:

```
... The blockquote element
    ...
    ... User agents may allow users to follow such citation links, but they are primarily
    intended for private use (e.g. by server-side scripts collecting statistics about a
    site's use of quotations), not for readers.

4.5.7. The q element
    ...
    ... User agents may allow users to follow such citation links, but they are primarily
    intended for private use (e.g. by server-side scripts collecting statistics about a
    site's use of quotations), not for readers.
```

***EdgeHTML Mode***

There is no way for the user to follow citation links.

## 2.2.12 [W3C-HTML52] Section 4.4.8. The li element

C0015: The maximum value of the value attribute is 2,147,483,647

The specification states:

```
4.4.7 The li element
    ...
    The value attribute, if present, must be a valid integer giving the ordinal value of
    the list item.
```

***EdgeHTML Mode***

The maximum value of the `value` attribute is 2,147,483,647. Any `li` element values that are larger are set to this maximum.

C0016: The minimum value of the value attribute is -2,147,483,648

The specification states:

```
4.4.7 The li element
    ...
    The value attribute, if present, must be a valid integer giving the ordinal value of
    the list item.
```

***EdgeHTML Mode***

The minimum value of the `value` attribute is -2,147,483,648. Any `li` element values that are smaller are set to this minimum.

## 2.2.13 [W3C-HTML52] Section 4.5.7. The q element

C0014: There is no way for the user to follow citation links

The specification states:

```
... The blockquote element
    ...
    ... User agents may allow users to follow such citation links, but they are primarily
    intended for private use (e.g. by server-side scripts collecting statistics about a
    site's use of quotations), not for readers.

4.5.7. The q element
    ...
    ... User agents may allow users to follow such citation links, but they are primarily
    intended for private use (e.g. by server-side scripts collecting statistics about a
    site's use of quotations), not for readers.
```

***EdgeHTML Mode***

There is no way for the user to follow citation links.

## 2.2.14 [W3C-HTML52] Section 4.6.3. Attributes common to ins and delelements

C0017: The datetime value is not shown to the user

The specification states:

```
4.6.3 Attributes common to ins and del elements
    ...
    The datetime attribute may be used to specify the time and date of the change.
    ...
    This value may be shown to the user, but it is primarily intended for private use.
```

***EdgeHTML Mode***

The `datetime` value is not shown to the user.

C0018: No way is provided for the user to follow citation links

The specification states:

```
4.6.3 Attributes common to ins and del elements
    ...
    If the cite attribute is present, it must be a valid URL potentially surrounded by
    spaces that explains the change.  ... User agents may allow users to follow such
    citation links, but they are primarily intended for private use (e.g. by server-side
    scripts collecting statistics about a site's edits), not for readers.
```

***EdgeHTML Mode***

No way is provided for the user to follow citation links.

## 2.2.15 [W3C-HTML52] Section 4.7.5. The img element

C0019: Images are obtained immediately

The specification states:

```
... The img element
    ...
    In a browsing context where scripting is disabled, user agents may obtain images
    immediately or on demand. ...
```

***EdgeHTML Mode***

Images are obtained immediately.

C0021: No image indicator is shown when the image is unavailable

The specification states:

```
... The img element
    ...
    What an img element represents depends on the src attribute and the alt attribute.

        If the src attribute is set and the alt attribute is set to the empty string
            ...
            ... User agents may provide the user with a notification that an image is
            present but has been omitted from the rendering.
```

***EdgeHTML Mode***

No image indicator is shown when the image is unavailable. If alt text is available that text will be shown.

### 2.2.16 [W3C-HTML52] Section 4.7.7. The embed element

C0022: The user is not provided an option to override the sandbox and instantiate the plugin anyway

The specification states:

```
... The embed element
    ...
    ... The user agent may offer the user the option to override the sandbox and
    instantiate the plugin anyway; if the user invokes such an option, the user agent
    must act as if the conditions above did not apply for the purposes of this element.
```

#### *EdgeHTML Mode*

The user is not provided an option to override the sandbox and instantiate the plugin anyway.

### 2.2.17 [W3C-HTML52] Section 4.7.10. The video element

C0023: Visual indicators provide the state of the video

The specification states:

```
... The video element
    ...
    In addition to the above, the user agent may provide messages to the user (such as
    "buffering", "no video loaded", "error", or more detailed information) by overlaying
    text or icons on the video or other areas of the element's playback area, or in
    another appropriate manner.
```

#### *EdgeHTML Mode*

Visual indicators provide the state of the video.

C0024: No external link is provided if the video cannot be rendered

The specification states:

```
... The video element
    ...
    User agents that cannot render the video may instead make the element represent a
    link to an external video playback utility or to the video data itself.
```

#### *EdgeHTML Mode*

No external link is provided if the video cannot be rendered.

C0025: Videos can be played fullscreen

The specification states:

```
4.7.6 The video element
    ...
```

```
User agents may allow users to view the video content in manners more suitable to the
user (e.g. fullscreen or in an independent resizable window). ...
```

***EdgeHTML Mode***

Videos can be played fullscreen.

C0026: Fullscreen videos show controls and ignore the controls attribute

The specification states:

```
... The video element
    ...
    ... In such an independent context, however, user agents may make full user
    interfaces visible ... even if the controls attribute is absent.
```

***EdgeHTML Mode***

Fullscreen videos show controls and ignore the `controls` attribute.

C0027: Screensavers are not disabled for fullscreen videos

The specification states:

```
... The video element
    ...
    User agents may allow video playback to affect system features that could interfere
    with the user's experience; for example, user agents could disable screensavers while
    video playback is in progress.
```

***EdgeHTML Mode***

Screensavers are not disabled for fullscreen videos.

## 2.2.18 [W3C-HTML52] Section 4.7.13.5. Loading the media resource

C0030: The preload attribute causes preloading of resources

The specification states:

```
... Loading the media resource
    ...
    The preload attribute is intended to provide a hint to the user agent about what the
    author thinks will lead to the best user experience. The attribute may be ignored
    altogether, for example based on explicit user preferences or based on the available
    connectivity.
```

***EdgeHTML Mode***

The `preload` attribute causes preloading of resources.

C0031: Buffered data is discarded only if data becomes invalid

The specification states:

```
... Loading the media resource
    ...
    User agents may discard previously buffered data.
```

***EdgeHTML Mode***

Buffered data is discarded only if data becomes invalid.

### 2.2.19 [W3C-HTML52] Section 4.7.13.8. Playing the media resource

C0032: Pitch adjustments are made when the playback rate is not 1.0

The specification states:

```
... Playing the media resource
    ...
    ... If the effective playback rate is not 1.0, the user agent may apply pitch
    adjustments to the audio as necessary to render it faithfully.
```

***EdgeHTML Mode***

Pitch adjustments are made when the playback rate is not 1.0.

### 2.2.20 [W3C-HTML52] Section 4.8.2. Links created by a and area elements

C0037: The user is not given a choice whether to navigate the hyperlink or download the resource

The specification states:

```
... Links created by a and area elements
    ...
    When an a or area element's activation behavior is invoked, the user agent may allow
    the user to indicate a preference regarding whether the hyperlink is to be used for
    navigation or whether the resource it specifies is to be downloaded.
```

***EdgeHTML Mode***

The user is not given a choice whether to navigate the hyperlink or download the resource.

### 2.2.21 [W3C-HTML52] Section 4.10.5.1.5. E-mail state (type=email)

C0039: Punycode in a value is not properly converted to IDN

The specification states:

```
4.10.5.1.5 E-mail state (type=email)
    ...
    User agents may transform the values for display and editing.
    User agents should convert punycode in the domain labels of the value to
    Internationalized Domain Names in the display, and vice versa.
```

***EdgeHTML Mode***

Punycode in a value is not properly converted to IDN.

### 2.2.22 [W3C-HTML52] Section 4.10.5.1.17. File Upload state (type=file)

C0040: The accept attribute is used to filter the file selection from the file picker

The specification states:

```
... File Upload state (type=file)
    ...
    User agents may use the value of this attribute to display a more appropriate user
    interface than a generic file picker. ...
```

***EdgeHTML Mode***

The `accept` attribute is used to filter the file selection from the file picker.

### 2.2.23 [W3C-HTML52] Section 4.10.18.3. Limiting user input length: the maxlengthattribute

C0041: A negative maxlength value is treated as if it were 0

The specification states:

```
... Limiting user input length: the maxlength attribute
    ...
    If an element has its form control maxlength attribute specified, the attribute's
    value must be a valid non-negative integer. If the attribute is specified and
    applying the rules for parsing non-negative integers to its value results in a
    number, then that number is the element's maximum allowed value length. If the
    attribute is omitted or parsing its value results in an error, then there is no
    maximum allowed value length.
```

***EdgeHTML Mode***

A negative `maxlength` value is treated as if it were 0. No characters are accepted.

### 2.2.24 [W3C-HTML52] Section 4.10.18.7.2. Processing model

C0042: Control values are stored and previously stored values are offered to the user

The specification states:

```
...
    ...
    When an element's autofill field name is not "off", the user agent may store the
    control's value, and may offer previously stored values to the user.
```

***EdgeHTML Mode***

Control values are stored and previously stored values are offered to the user.

## 2.2.25 [W3C-HTML52] Section 4.10.20.2. Constraint validation

C0043: Constraint validation error reporting procedures

The specification states:

```
4.10.21.2 Constraint validation
    ...
    If a user agent is to interactively validate the constraints of form element form,
    then the user agent must run the following steps:
        ...
        3.  Report the problems with the constraints of at least one of the elements
            given in unhandled invalid controls to the user. User agents may focus one of
            those elements in the process, by running the focusing steps for that
            element, and may change the scrolling position of the document, or perform
            some other action that brings the element to the user's attention. User
            agents may report more than one constraint violation. User agents may
            coalesce related constraint violation reports if appropriate (e.g. if
            multiple radio buttons in a group are marked as required, only one error need
            be reported). If one of the controls is not being rendered (e.g. it has the
            hidden attribute set) then user agents may report a script error.
```

***EdgeHTML Mode***

Constraint validation error reporting procedures include:

- Reporting and marking all constraint violations on the form
- Placing red borders around the input fields
- Changing the scrolling position to the first violation

They do not include:

- Coalescing of related constraint violations
- Reporting of script errors

## 2.2.26 [W3C-HTML52] Section 4.10.21.7. Multipart form data

C0044: Form fields, including filename fields, are encoded in UTF-8 and are not approximated

The specification states:

```
... Multipart form data
    ...
    The multipart/form-data encoding algorithm is as follows:
```

```
...
5.  ...
    File names included in the generated multipart/form-data resource (as part of
    file fields) must use the character encoding selected above, though the
    precise name may be approximated if necessary (e.g. newlines could be removed
    from file names, quotes could be changed to "%22", and characters not
    expressible in the selected character encoding could be replaced by other
    characters). ...
```

***EdgeHTML Mode***

Form fields, including filename fields, are encoded in UTF-8 and are not approximated.

## 2.2.27 [W3C-HTML52] Section 4.12.4.2. Serializing bitmaps to a file

C0045: Many image formats other than PNG are supported

The specification states:

```
... Serializing bitmaps to a file
    ...
    User agents must support PNG ("image/png"). User agents may support other types. If
    the user agent does not support the requested type, it must create the file using the
    PNG format.
```

***EdgeHTML Mode***

Many image formats other than PNG are supported (gif, jpeg, ico, bmp, etc.).

## 2.2.28 [W3C-HTML52] Section 5.1. The hidden attribute

C0070: Assistive technologies determine what is done with the hidden items

The specification states:

```
... The hidden attribute
    ...
    When such features are available, User Agents may use them to expose the full
    semantics of hidden elements to AT when appropriate, if such content is referenced
    indirectly by an ID reference or valid hash-name reference. This allows ATs to access
    the structure of these hidden elements upon user request, while keeping the content
    hidden in all presentations of the normal document flow. Authors who wish to prevent
    user-initiated viewing of a hidden element should not reference the element with such
    a mechanism.
```

***EdgeHTML Mode***

Assistive technologies have access to elements that are in the hidden state, and those technologies determine what is done with the hidden items.

### 2.2.29 [W3C-HTML52] Section 5.2. Inert subtrees

C0072: Selection and find on a page are prevented from working when the page is inert because of a dialog

The specification states:

```
... Inert subtrees

    ... When a node is inert, then the user agent must act as if the node was absent for
    the purposes of targeting user interaction events, may ignore the node for the
    purposes of text search user interfaces (commonly known as "find in page"), and may
    prevent the user from selecting text in that node. ...
```

***EdgeHTML Mode***

Selection and find on a page are prevented from working when the page is inert because of a dialog.

### 2.2.30 [W3C-HTML52] Section 5.4.2. Data model

C0086: Focusable elements follow the platform conventions for accessibility

The specification states:

```
5.4.2. Data model

    The term focusable area is used to refer to regions of the interface that can become
    the target of keyboard input. Focusable areas can be elements, parts of elements, or
    other regions managed by the user agent.
    ...
    The following table describes what objects can be focusable areas. ...
        Focusable area [column]
            ...
            Any other element or part of an element, especially to aid with accessibility
            or to better match platform conventions.
```

***EdgeHTML Mode***

Focusable elements follow the platform conventions for accessibility.

### 2.2.31 [W3C-HTML52] Section 5.4.6. Focus management APIs

C0074: The blur function is not ignored on elements but is ignored on the window object

The specification states:

```
... ...
    ...
    The blur() method, when invoked, should run the unfocusing steps for the element on
    which the method was called ... . User agents may selectively or uniformly ignore
    calls to this method for usability reasons.
```

***EdgeHTML Mode***

The `blur` function is not ignored on elements but is ignored on the `window` object.

## 2.2.32 [W3C-HTML52] Section 5.6.5. Spelling and grammar checking

C0077: The lang attribute defined on an element determines the spellcheck language

The specification states:

```
... Spelling and grammar checking
    ...
    If the checking is enabled for a word/sentence/text, the user agent should indicate
    spelling and grammar errors in that text. User agents should take into account the
    other semantics given in the document when suggesting spelling and grammar
    corrections. User agents may use the language of the element to determine what
    spelling and grammar rules to use, or may use the user's preferred language settings.
    UAs should use input element attributes such as pattern to ensure that the resulting
    value is valid, where possible.
```

***EdgeHTML Mode***

The `lang` attribute defined on an element determines the spellcheck language.

C0078: Spelling and grammar errors on the text preloaded with the page are not reported

The specification states:

```
... Spelling and grammar checking
    ...
    Even when checking is enabled, user agents may opt to not report spelling or grammar
    errors in text that the user agent deems the user has no interest in having checked
    (e.g. text that was already present when the page was loaded, or that the user did
    not type, or text in controls that the user has not focused, or in parts of e-mail
    addresses that the user agent is not confident were misspelt).
```

***EdgeHTML Mode***

Spelling and grammar errors on the text preloaded with the page are not reported.

## 2.2.33 [W3C-HTML52] Section 6.1.5. Browsing context names

C0048: If the sandboxed auxiliary navigation browsing context flag is set, a new browsing context is created

The specification states:

```
... Browsing context names
    ...
    The rules for choosing a browsing context given a browsing context name are as
    follows. The rules assume that they are being applied in the context of a browsing
    context, as part of the execution of a task.
        ...
        5.  Otherwise, a new browsing context is being requested, and what happens
            depends on the user agent's configuration and abilities — it is determined by
            the rules given for the first applicable option from the following list:
```

```
...
If the current browsing context's active document's active sandboxing
flag set has the sandboxed auxiliary navigation browsing context flag set.

    Typically, there is no chosen browsing context.

    The user agent may offer to create a new top-level browsing context
    or reuse an existing top-level browsing context. ...
```

***EdgeHTML Mode***

If the active sandboxing flag set of the current browsing context's active document has the sandboxed auxiliary navigation browsing context flag set, a new browsing context is created.

## 2.2.34 [W3C-HTML52] Section 6.6.1. The session history of browsing contexts

C0050: Document objects are discarded based on content expiration, disk space usage, and preferences for content storage

The specification states:

```
... ...
    ...
    User agents may discard the Document objects of entries other than the current entry
    that are not referenced from any script, reloading the pages afresh when the user or
    script navigates back to such pages. This specification does not specify when user
    agents should discard Document objects and when they should cache them.
```

***EdgeHTML Mode***

Document objects are discarded based on content expiration, disk space usage, and preferences for content storage.

## 2.2.35 [W3C-HTML52] Section 6.6.2. The History interface

C0051: The maximum number of state objects added to the session history for a page is 1,048,576

The specification states:

```
... The History interface
    ...
    User agents may limit the number of state objects added to the session history per
    page. If a page hits the UA-defined limit, user agents must remove the entry
    immediately after the first entry for that Document object in the session history
    after having added the new entry. (Thus the state history acts as a FIFO buffer for
    eviction, but as a LIFO buffer for navigation.)
```

***EdgeHTML Mode***

The maximum number of state objects added to the session history for a page is 1,048,576.

### 2.2.36 [W3C-HTML52] Section 6.7.1. Navigating across documents

C0053: Navigation errors are shown for all document response codes other than code value 200

The specification states:

```
... Navigating across documents
    ...
    When a browsing context is navigated to a new resource, the user agent must run the
    following steps:
        ...
        17. ...
            ... The user agent may indicate to the user that the navigation has been
            aborted for security reasons.
```

***EdgeHTML Mode***

Navigation errors are shown for all document response codes other than code value 200.

### 2.2.37 [W3C-HTML52] Section 6.7.3. Page load processing model for XML files

C0054: The root element performs a namespace-based lookup in order to determine if the content is a feed

The specification states:

```
... Page load processing model for XML files
    ...
    User agents may examine the namespace of the root Element node of this Document
    object to perform namespace-based dispatch to alternative processing tools, e.g.
    determining that the content is actually a syndication feed and passing it to a feed
    handler. If such processing is to take place, abort the steps in this section, and
    jump to the next step (labeled non-document content) in the navigate steps above.
```

***EdgeHTML Mode***

The root element performs a namespace-based lookup in order to determine if the content is a feed.

### 2.2.38 [W3C-HTML52] Section 6.7.4. Page load processing model for text files

C0055: No content is added to the head element of the document

The specification states:

```
... Page load processing model for text files
    ...
    User agents may add content to the head element of the Document, ...
```

***EdgeHTML Mode***

No content is added to the head element of the document.

### 2.2.39 [W3C-HTML52] Section 6.7.6. Page load processing model for media

C0056: A head section is added to the content of a Document

The specification states:

```
... Page load processing model for media
    ...
    User agents may add content to the head element of the Document, or attributes to the
    element host element, e.g. to link to a style sheet ..., give the document a title,
    make the media autoplay, etc.
```

#### *EdgeHTML Mode*

A `head` section is added to the content of a `Document`.

### 2.2.40 [W3C-HTML52] Section 6.7.10. History traversal

C0057: The scroll state is retained for back and forward navigations

The specification states:

```
... History traversal
    ...
    When a user agent is required to traverse the history to a specified entry,
    optionally with replacement enabled, and optionally with the
    [non-blocking/asynchronous] events flag set, the user agent must act as follows.
        ...
        9.  If the entry is an entry with persisted user state, the user agent may ...
            update aspects of the document and its rendering ... .
```

#### *EdgeHTML Mode*

The scroll state is retained for back and forward navigations.

### 2.2.41 [W3C-HTML52] Section 6.7.11. Unloading documents

C0058: The prompt does not show the returnValue

The specification states:

```
... Unloading documents
    ...
    When a user agent is to prompt to unload a document, it must run the following steps.
        1. …
        ...  ...
            The prompt shown by the user agent may include the string of the returnValue
            attribute, ...
```

#### *EdgeHTML Mode*

The prompt does not show the `returnValue`.

### 2.2.42 [W3C-HTML52] Section 6.7.12. Aborting a document load

C0059: A user can invoke the abort a document algorithm by clicking the stop button in the address bar

The specification states:

```
... Aborting a document load
    ...
    User agents may allow users to explicitly invoke the abort a document algorithm for a
    Document. ...
```

***EdgeHTML Mode***

A user can invoke the abort a document algorithm by clicking the stop button in the address bar.

### 2.2.43 [W3C-HTML52] Section 7.1.2. Enabling and disabling scripting

C0063: The user can set a preference to disable scripting

The specification states:

```
... Enabling and disabling scripting

    Scripting is enabled in a browsing context when all of the following conditions are
    true:
        ...
        The user has not disabled scripting for this browsing context at this time. (User
        agents may provide users with the option to disable scripting globally, or in a
        finer-grained manner, e.g. on a per-origin basis.)
```

***EdgeHTML Mode***

The user can set a preference to disable scripting.

### 2.2.44 [W3C-HTML52] Section 7.1.5.1. Event handlers

C0068: An unparsable body results in an error reported to the user

The specification states:

```
... Event handlers
    ...
    When the user agent is to get the current value of the event handler H, it must run
    these steps:

        1.  If H's value is an internal raw uncompiled handler, run these substeps:
            ...
            ...  If body is not parsable as FunctionBody or if parsing detects an early
            error, then follow these substeps:
                ...
                2.  Report the error for the appropriate script and with the appropriate
                    position (line number and column number) given by location, using the
                    global object specified by script settings as the target. If the
                    error is still not handled after this, then the error may be reported
```

```
... .
```

***EdgeHTML Mode***

An unparsable body results in an error reported to the user.

## 2.2.45 [W3C-HTML52] Section 7.6.2. Printing

C0069: Printing events do not wait for the user to accept or decline

The specification states:

```
... Printing
    ...
    The printing steps are as follows:
        ...
        4.  The user agent should offer the user the opportunity to obtain a physical
            form (or the representation of a physical form) of the document. The user
            agent may wait for the user to either accept or decline before returning; if
            so, the user agent must pause while the method is waiting. Even if the user
            agent doesn't wait at this point, the user agent must use the state of the
            relevant documents as they are at this point in the algorithm if and when it
            eventually creates the alternate form.
```

***EdgeHTML Mode***

Printing events do not wait for the user to accept or decline.

## 2.2.46 [W3C-HTML52] Section 8.2. Parsing HTML documents

C0079: Parsing continues even if there are parsing errors

The specification states:

```
8.2 Parsing HTML documents
    ...
    This specification defines the parsing rules for HTML documents, whether they are
    syntactically correct or not. Certain points in the parsing algorithm are said to be
    parse errors. The error handling for parse errors is well-defined (that's the
    processing rules described throughout this specification), but user agents, while
    parsing an HTML document, may abort the parser at the first parse error that they
    encounter for which they do not wish to apply the rules described in this
    specification.
```

***EdgeHTML Mode***

Parsing continues even if there are parsing errors. The errors are reported to the console. An abort does not occur unless there is a catastrophic failure.

## 2.2.47 [W3C-HTML52] Section 8.2.7. Coercing an HTML DOM into an infoset

C0080: Attributes are dropped if they start with xmlns in the case of no namespace

The specification states:

```
8.2.7 Coercing an HTML DOM into an infoset
    ...
    If the XML API doesn't support attributes in no namespace that are named "xmlns",
    attributes whose names start with "xmlns:", or attributes in the XMLNS namespace,
    then the tool may drop such attributes.
```

***EdgeHTML Mode***

Attributes are dropped if they start with `xmlns:` in the case of no namespace.

C0081: Local names of elements and attributes are limited to the ASCII character range

The specification states:

```
8.2.7 Coercing an HTML DOM into an infoset
    ...
    If the XML API being used restricts the allowable characters in the local names of
    elements and attributes, then the tool may map all element and attribute local names
    that the API wouldn't support to a set of names that are allowed, by replacing any
    character that isn't supported with the uppercase letter U and the six digits of the
    character's Unicode code point when expressed in hexadecimal, using digits 0-9 and
    capital letters A-F as the symbols, in increasing numeric order.
```

***EdgeHTML Mode***

Local names of elements and attributes are limited to the ASCII character range.

C0082: No space is inserted between consecutive "-" (U+002D) characters or after one that ends a line

The specification states:

```
8.2.7 Coercing an HTML DOM into an infoset
    ...
    If the XML API restricts comments from having two consecutive U+002D HYPHEN-MINUS
    characters (--), the tool may insert a single U+0020 SPACE character between any such
    offending characters.

    If the XML API restricts comments from ending in a U+002D HYPHEN-MINUS character (-),
    the tool may insert a single U+0020 SPACE character at the end of such comments.
```

***EdgeHTML Mode***

No space is inserted between consecutive "-" (U+002D) characters or after one that ends a line.

## 2.2.48 [W3C-HTML52] Section 9.3. Serializing XML fragments

C0083: When XHTML documents are serialized, prefixes and namespace declarations are adjusted as needed

The specification states:

```
9.3 Serializing XHTML fragments
    ...
    In both cases, the string returned must be XML namespace-well-formed and must be an
    isomorphic serialization of all of that node's relevant child nodes, in tree order.
    User agents may adjust prefixes and namespace declarations in the serialization (and
    indeed might be forced to do so in some cases to obtain namespace-well-formed XML).
    User agents may use a combination of regular text and character references to
    represent Text nodes in the DOM.
```

***EdgeHTML Mode***

When XHTML documents are serialized, prefixes and namespace declarations are adjusted as needed.

### 2.2.49 [W3C-HTML52] Section 11.3.4. Other elements, attributes and APIs

C0084: The scheme attribute is not used as an extension of the name attribute

The specification states:

```
... Other elements, attributes and APIs
    ...
    User agents may treat the scheme content attribute on the meta element as an
    extension of the element's name content attribute when processing a meta element with
    a name attribute whose value is one that the user agent recognizes as supporting the
    scheme attribute.
```

***EdgeHTML Mode***

The `scheme` attribute is not used as an extension of the `name` attribute.

## 2.3 Extensions

There are no extensions to the requirements of [W3C-HTML52].

### 2.3.1 [W3C-HTML52] Section 5.6.2. Making entire documents editable: The designMode IDL attribute

E0001: If the designMode IDL attribute matches the value “inherit” then designMode is enabled if the parent designMode is enabled

The specification states:

```
7.6.2 Making entire documents editable: The designMode IDL attribute
    ...
    The designMode IDL attribute on the Document object takes two values, "on" and "off".
    On setting, the new value must be compared in an ASCII case-insensitive manner to
    these two values; if it matches the "on" value, then designMode must be enabled, and
    if it matches the "off" value, then designMode must be disabled. Other values must be
    ignored.
```

***EdgeHTML Mode***

If the `designMode` IDL attribute matches the value "inherit" then `designMode` is enabled if the parent `designMode` is enabled and disabled if the parent `designMode` is disabled.

## 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