[MS-HTML5]:

Microsoft Edge / Internet Explorer HTML5 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.
- **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.

Revision Summary

Date	Revision History	Revision Class	Comments
4/17/2015	1.0	New	Released new document
7/7/2015	1.1	Minor	Clarified the meaning of the technical content.
11/2/2015	1.2	Minor	Clarified the meaning of the technical content.
3/22/2016	1.3	Minor	Clarified the meaning of the technical content.
7/19/2016	1.4	Minor	Clarified the meaning of the technical content.
11/2/2016	1.4	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document describes the level of support provided by Microsoft web browsers for the *HTML5 Specification* [HTML5] W3C Recommendation of 28 October 2014.

The [HTML5] specification contains guidance for authors of webpages and web apps, in addition to authoring tools, conformance checking tools and user agents (browser applications). Statements found in this document apply only to normative requirements in the specification targeted to user agents.

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.

[HTML5] Berjon, R., Faulkner, S., Leithead, T., Navara, E., et al., Eds., "HTML5 -- A vocabulary and associated APIs for HTML and XHTML", http://www.w3.org/TR/html5/

[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

1.2.2 Informative References

None.

1.3 Microsoft Implementations

The following Microsoft web browsers implement some portion of the [HTML5] specification:

- Windows Internet Explorer 9
- Windows Internet Explorer 10
- Internet Explorer 11
- Internet Explorer 11 for Windows 10
- Microsoft Edge

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

Browser Version	Document Modes Supported
Internet Explorer 9	Quirks Mode IE7 Mode IE8 Mode IE9 Mode
Internet Explorer 10	Quirks Mode IE7 Mode IE8 Mode IE9 Mode IE10 Mode
Internet Explorer 11	Quirks Mode IE7 Mode IE8 Mode IE9 Mode IE10 Mode IE11 Mode
Internet Explorer 11 for Windows 10	Quirks Mode IE7 Mode IE8 Mode IE9 Mode IE10 Mode IE11 Mode
Microsoft Edge	EdgeHTML Mode

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

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

1.4 Standards Support Requirements

To conform to [HTML5], 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 [HTML5] 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 a full list of variations, clarifications, and extension points in the Microsoft implementation of [HTML5].

- Section 2.1 includes only those variations that violate a MUST requirement in the target specification.
- Section 2.2 describes further variations from MAY and SHOULD requirements.
- Section 2.3 identifies variations in error handling.
- Section 2.4 identifies variations that impact security.

2.1 Normative Variations

The following subsections detail the normative variations from MUST requirements in [HTML5].

2.1.1 [HTML5] 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]
```

All document modes (All versions)

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 [HTML5] Section 2.4.4.3 Floating-point numbers

V0002: 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) character.
        2.  One or both of the following, in the given order:
            1.  A series of one or more ASCII digits.
            2.
                1.  A single "." (U+002E) character.
                2.  A series of one or more ASCII digits.
        3. Optionally:
            1.  Either a "e" (U+0065) character or a "E" (U+0045) character.
            2.  Optionally, a "-" (U+002D) character or "+" (U+002B) character.
            3.  A series of one or more ASCII digits.
```

All document modes (All versions)

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

V0003: A number ending with "." is not reverted to ""

The specification states:

```
2.4.4.3 Floating-point numbers
    ...
    The rules for parsing floating-point number values are as given in the following
    algorithm. This algorithm must be aborted at the first step that returns something.
    This algorithm will return either a number or an error.
        ...
        9.  If the character indicated by position is a "." (U+002E), and that is not the
            last character in input, and the character after the character indicated by
            position is an ASCII digit, then set value to zero and jump to the step
            labeled fraction.
```

All document modes (All versions)

A number ending with "." (U+002E) is not reverted to the empty string ("").

V0004: Multiple decimal points in a number are considered valid

The specification states:

```
2.4.4.3 Floating-point numbers
    ...
    The rules for parsing floating-point number values are as given in the following
    algorithm. This algorithm must be aborted at the first step that returns something.
    This algorithm will return either a number or an error.
        ...
        9.  If the character indicated by position is a "." (U+002E), and that is not the
            last character in input, and the character after the character indicated by
            position is an ASCII digit, then set value to zero and jump to the step
            labeled fraction.
        10. If the character indicated by position is not an ASCII digit, then return an
            error.
```

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

Multiple decimal points in a number are considered valid. All but the first are ignored.

V0005: Missing number or sign after "e" or "E" is considered a valid 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) character.
    2.  One or both of the following, in the given order:
        1.  A series of one or more ASCII digits.
```

```
        2.
            1.  A single "." (U+002E) character.
            2.  A series of one or more ASCII digits.
    3. Optionally:
        1.  Either a "e" (U+0065) character or a "E" (U+0045) character.
        2.  Optionally, a "-" (U+002D) character or "+" (U+002B) character.
        3.  A series of one or more ASCII digits.
```

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

The absence of a number after "e" or "E", or optional "-" (U+002D) or "+" (U+002B), is considered a valid floating point number. If there is no sign after the "E" or "e", "+" is assumed. If there is no number, 0 is assumed.

2.1.3 [HTML5] Section 2.5.3 Dynamic changes to base URLs

V0006: Changes to the base href value do not update the URLs

The specification states:

```
2.5.3 Dynamic changes to base URLs

    When an xml:base attribute is set, changed, or removed, the attribute's element, and
    all descendant elements, are affected by a base URL change.

    When a document's document base URL changes, all elements in that document are
    affected by a base URL change.
```

All document modes (All versions)

The document URLs do not change when the `href` values for `xml:base` or the `base` element change.

2.1.4 [HTML5] Section 2.7.2.1 HTMLAllCollection

V0007: The tags attribute is not supported

The specification states:

```
2.7.2.1 HTMLAllCollection
    ...
    interface HTMLAllCollection : HTMLCollection {
        ...
        HTMLAllCollection tags(DOMString tagName);
    };
```

All document modes (All versions)

The `tags` attribute is not supported.

V0008: The item function does not return an HTMLCollection

The specification states:

```
2.7.2.1 HTMLAllCollection

    interface HTMLAllCollection : HTMLCollection {
        ...
        (HTMLCollection or Element)? item(DOMString name);
        ...
    };
```

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

The `item` function does not return an `HTMLCollection` if there are multiple matching items and only returns a single `Element`.

V0009: The namedItem function does not return an HTMLCollection

The specification states:

```
2.7.2.1 HTMLAllCollection
    ...
    interface HTMLAllCollection : HTMLCollection {
        ...
        legacycaller getter (HTMLCollection or Element)? namedItem(DOMString name);
         ...
    };
```

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

The `namedItem` function does not return an `HTMLCollection` if there are multiple matching items and only returns a single `Element`.

V0010: The item attribute is not specified on the HTMLAllCollection instance

The specification states:

```
2.7.2.1 HTMLAllCollection
    ...
    interface HTMLAllCollection : HTMLCollection {
        ...
        (HTMLCollection or Element)? item(DOMString name);
        ...
    };
```

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

The `item` attribute is not specified on the `HTMLAllCollection` instance but is defined on the interface `HTMLCollection`.

V0011: The HTMLAllCollection interface is not defined

The specification states:

```
2.7.2.1 HTMLAllCollection
    ...
    interface HTMLAllCollection : HTMLCollection {
        ...
    };
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `HTMLAllCollection` interface is not defined.

2.1.5 [HTML5] Section 2.7.2.2 HTMLFormControlsCollection

V0012: The HTMLFormControlsCollection interface is not supported

The specification states:

```
2.7.2.2 HTMLFormControlsCollection

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

    interface HTMLFormControlsCollection : HTMLCollection {
        ...
    };
```

All document modes (All versions)

The `HTMLFormControlsCollection` interface is not supported.

2.1.6 [HTML5] Section 2.7.2.3 HTMLOptionsCollection

V0013: The HTMLOptionsCollection interface is not supported

The specification states:

```
2.7.2.3 HTMLOptionsCollection

    The HTMLOptionsCollection interface is used for collections of option elements. It is
    always rooted on a select element and has attributes and methods that manipulate that
    element's descendants.

    interface HTMLOptionsCollection : HTMLCollection {
        ...
    };
```

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

The `HTMLOptionsCollection` interface is not supported.

2.1.7 [HTML5] Section 2.7.6 Callbacks

V0014: The FileCallback callback is not supported

The specification states:

```
2.7.6 Callbacks
    ...
    The following callback function type is used in various APIs that interact with File
    objects:

        callback FileCallback = void (File file);
```

All document modes (All versions)

The `FileCallback` callback is not supported.

2.1.8 [HTML5] Section 3.1.1 The Document object

V0015: The readyState attribute is type DOMString, not DocumentReadyState.

The specification states:

```
3.1.1 The Document object
    ...
    partial /*sealed*/ interface Document {
        ...
        readonly attribute DocumentReadyState readyState;
        ...
    };
```

All document modes (All versions)

The `readyState` attribute is type `DOMString`, not `DocumentReadyState`.

2.1.9 [HTML5] Section 3.1.2 Resource metadata management

V0016: A cookie-averse Document object does not raise a SecurityError when origin is not a correct tuple

The specification states:

```
3.1.2 Resource metadata management
    ...
    On getting, if the document is a cookie-averse Document object, then the user agent
    must return the empty string. Otherwise, if the Document's origin is not a
    scheme/host/port tuple, the user agent must throw a SecurityError exception.
```

All document modes (All versions)

A cookie-averse `Document` object does not throw a `SecurityError` when origin is not a scheme/host/port tuple and no exception is thrown.

2.1.10 [HTML5] 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;
        ...
    };
```

All document modes (All versions)

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;
        ...
    };
```

All document modes (All versions)

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

2.1.11 [HTML5] Section 3.2.4.1.7 Interactive content

V0019: A synthetic click does not correctly set the isTrusted flag

The specification states:

```
3.2.4.1.7 Interactive content
    ...
    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.
```

All document modes (All versions)

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

2.1.12 [HTML5] Section 3.2.5.4 The translate attribute

V0020: The translate attribute is not supported

The specification states:

```
3.2.5.4 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.
```

All document modes (All versions)

The `translate` attribute is not supported.

2.1.13 [HTML5] Section 3.2.5.6 The dir attribute

V0021: The auto keyword is not supported

The specification states:

```
3.2.5.6 The dir attribute
    ...
    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).
```

All document modes (All versions)

The `auto` keyword is not supported.

2.1.14 [HTML5] Section 3.2.5.9 Embedding custom non-visible data with the data-* attributes

V0022: A data- attribute that contains an uppercase letter after the dash is not properly set

The specification states:

```
3.2.5.9 Embedding custom non-visible data with the data-* attributes
    ...
    The algorithm for setting names to certain values
        ...
        4. For each uppercase ASCII letter in name, insert a "-" (U+002D) character
           before the character and replace the character with the same character
           converted to ASCII lowercase.
```

All document modes (All versions)

A `data-` attribute that contains an uppercase letter after the dash does not set itself with a dash before the lowercasing the letter.

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

The specification states:

```
3.2.5.9 Embedding custom non-visible data with the data-* attributes

    The algorithm for setting names to certain values
        ...
        3. If name contains a "-" (U+002D) character followed by a lowercase ASCII
           letter, throw a SyntaxError exception and abort these steps.
```

All document modes (All versions)

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

2.1.15 [HTML5] 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.
```

All document modes (All versions)

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 normalize the returned string

The specification states:

```
4.2.2 The title element
    ...
    The IDL attribute text must return a concatenation of the contents of all the Text
    nodes that are children of the title element (ignoring any other nodes such as
    comments or elements), in tree order.
```

All document modes (All versions)

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

2.1.16 [HTML5] Section 4.2.3 The base element

V0026: When href of HTMLBaseElement is the empty string, it returns the empty string, not the full URL

The specification states:

```
4.2.3 The base element
    ...
    The href IDL attribute, on getting, must return the result of running the following
    algorithm:

        1.  If the base element has no href content attribute, then return the document
            base URL and abort these steps.
        2.  Let fallback base url be the Document's fallback base URL.
        3.  Let url be the value of the href attribute of the base element.
        4.  Resolve url relative to fallback base url (thus, the base href attribute
            isn't affected by xml:base attributes or base elements).
        5.  If the previous step was successful, return the resulting absolute URL and
            abort these steps.
        6.  Otherwise, return the empty string.
```

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

When `href` of `HTMLBaseElement` is the empty string, it returns the empty string (""), not the full URL.

V0027: When href of HTMLBaseElement does not exist, it returns the empty string, not the full URL

The specification states:

```
4.2.3 The base element
    ...
    The href IDL attribute, on getting, must return the result of running the following
    algorithm:
        ...
        1.  If the base element has no href content attribute, then return the document
            base URL and abort these steps.
        ...
```

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

When the `href` attribute of `HTMLBaseElement` does not exist, it returns the empty string (""), not the full URL.

2.1.17 [HTML5] Section 4.2.4 The link element

V0028: The crossOrigin attribute is not supported

The specification states:

```
4.2.4 The link element
    ...
    The crossorigin attribute is a CORS settings attribute. It is intended for use with
```

```
external resource links.
```

IE10 mode and IE9 mode (All versions)

The `crossorigin` attribute is not supported.

V0029: The sizes attribute is not supported

The specification states:

```
4.2.4 The link element
    ...
    The sizes attribute is used with the icon link type. ...
```

All document modes (All versions)

The `sizes` attribute is not supported.

V0030: The relList attribute is not supported

The specification states:

```
4.2.4 The link element
    ...
    interface HTMLLinkElement : HTMLElement {
        ...
        readonly attribute DOMTokenList relList;
        ...
    };
```

All document modes (All versions)

The `relList` attribute is not supported.

2.1.18 [HTML5] Section 4.2.6 The style element

V0031: The disabled attribute is not supported

The specification states:

```
4.2.6 The style element
    ...
    The disabled IDL attribute behaves as defined for the alternative style sheets DOM.
```

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

The `disabled` attribute is not supported.

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.
```

All document modes (All versions)

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

2.1.19 [HTML5] 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.
```

All document modes (All versions)

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

V0034: The onscroll 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.
```

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

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

V0035: The onpagehide and onpageshow event handlers are not supported

The specification states:

```
4.3.1 The body element
    ...
    Content attributes:
        ...
        onpagehide
        onpageshow
        ...
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `onpagehide` and `onpageshow` event handlers are not supported.

V0036: The onpopstate event handler is not supported

The specification states:

```
4.3.1 The body element
    ...
    Content attributes:
        ...
        onpopstate
        ...
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `onpopstate` event handler is not supported.

2.1.20 [HTML5] Section 4.3.10.1 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.
```

All document modes (All versions)

There is no graphical outline mechanism.

2.1.21 [HTML5] Section 4.4.5 The ol element

V0038: The reversed attribute is not supported

The specification states:

```
4.4.5 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, ...).
```

All document modes (All versions)

The `reversed` attribute is not supported.

2.1.22 [HTML5] Section 4.5.1 The a element

V0039: The download attribute is not supported

The specification states:

```
4.5.1 The a element
    ...
    interface HTMLAnchorElement : HTMLElement {
        ...
        attribute DOMString download;
        ...
    };
```

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

The `download` attribute is not supported.

V0040: The relList attribute is not supported

The specification states:

```
4.5.1 The a element
    ...
    interface HTMLAnchorElement : HTMLElement {
        ...
       readonly attribute DOMTokenList relList;
       ...
    };
```

All document modes (All versions)

The `relList` attribute is not supported.

V0041: The URLUtils interface is not implemented for the HTMLAnchorElement interface

The specification states:

```
4.5.1 The a element
    ...
    HTMLAnchorElement implements URLUtils;
```

All document modes (All versions)

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

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

`href`

`protocol`

`host`

`hostname`

`port`

`pathname`

`search`

`hash`

These are not implemented:

`username`

`password`

`searchParams`

`origin`

2.1.23 [HTML5] Section 4.5.10 The data element

V0042: The data element is not supported

The specification states:

```
4.5.10 The data element
    ...
    interface HTMLDataElement : HTMLElement {
        ...
    };
```

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

The `data` element is not supported.

2.1.24 [HTML5] Section 4.5.11 The time element

V0043: The time element is not supported

The specification states:

```
4.5.11 The time element
    ...
    interface HTMLTimeElement : HTMLElement {
        ...
```

```
};
```

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

The `time` element is not supported.

2.1.25 [HTML5] Section 4.5.22 The rb element

V0044: An rb element with no end tag does not properly close

The specification states:

```
4.5.22 The rb element
    ...
    An rb element's end tag may be omitted if the rb element is immediately followed by
    an rb, rt, rtc or rp element, or if there is no more content in the parent element.
```

All document modes (All versions)

An `rb` element with no end tag does not properly close when immediately followed by any of the following elements `rb`, `rt`, `rtc`, `rp`.

2.1.26 [HTML5] Section 4.5.24 The rtc element

V0045: The rtc element is not supported

The specification states:

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

All document modes (All versions)

The `rtc` element is not supported.

2.1.27 [HTML5] 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.
```

All document modes (All versions)

The `bdi` element is not supported.

2.1.28 [HTML5] Section 4.7.1 The img element

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

The specification states:

```
4.7.1 The img element
    ...
    ... User agents must only display the first page of a multipage resource (e.g. a PDF
    file). ...
```

All document modes (All versions)

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

2.1.29 [HTML5] Section 4.7.2 The iframe element

V0049: The srcdoc attribute is not supported

The specification states:

```
4.7.2 The iframe element
    ...
    interface HTMLIFrameElement : HTMLElement {
        ...
        attribute DOMString srcdoc;
        ...
    };
```

All document modes (All versions)

The `srcdoc` attribute is not supported.

2.1.30 [HTML5] Section 4.7.3 The embed element

V0050: The type attribute is not supported

The specification states:

```
4.7.3 The embed element

    interface HTMLEmbedElement : HTMLElement {
        ...
        attribute DOMString type;
        ...
    };
```

All document modes (All versions)

The `type` attribute is not supported.

2.1.31 [HTML5] Section 4.7.4 The object element

V0051: The typeMustMatch attribute is not supported

The specification states:

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

All document modes (All versions)

The `typeMustMatch` attribute is not supported.

V0052: The contentWindow attribute is not supported

The specification states:

```
4.7.4 The object element
    ...
    interface HTMLObjectElement : HTMLElement {
        ...
        readonly attribute WindowProxy? contentWindow;
        ...
    };
```

All document modes (All versions)

The `contentWindow` attribute is not supported.

2.1.32 [HTML5] Section 4.7.6 The video element

V0053: The crossorigin attribute is not supported

The specification states:

```
4.7.6 The video element
    ...
    Content attributes:
        Global attributes
        ...
        crossorigin – How the element handles crossorigin requests
        ...
```

All document modes (All versions)

The `crossorigin` attribute is not supported.

2.1.33 [HTML5] Section 4.7.8 The source element

V0054: Appending a source element using appendChild does not invoke the resource selection algorithm

The specification states:

```
4.7.8 The source element
    ...
    If a source element is inserted as a child of a media element that has no src
    attribute and whose networkState has the value NETWORK EMPTY, the user agent must
    invoke the media element's resource selection algorithm.
```

All document modes (All versions)

Appending a `source` element using `appendChild` does not invoke the resource selection algorithm when the element is appended.

2.1.34 [HTML5] Section 4.7.9 The track element

V0055: The track element is not supported

The specification states:

```
4.7.9 The track element
    ...
    interface HTMLTrackElement : HTMLElement {
        ...
    };
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `track` element is not supported.

V0056: The five unsigned short values of HTMLTrackElement are not supported

The specification states:

```
4.7.9 The track element
    ...
    interface HTMLTrackElement : HTMLElement {
        ...
        const unsigned short NONE = 0;
        const unsigned short LOADING = 1;
        const unsigned short LOADED = 2;
        const unsigned short ERROR = 3;
        readonly attribute unsigned short readyState;
        ...
```

```
    };
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The five unsigned short values `NONE`, `LOADING`, `LOADED`, `ERROR`, and `readyState` are not supported.

2.1.35 [HTML5] Section 4.7.10 Media elements

V0057: The crossOrigin attribute is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        attribute DOMString crossOrigin;
        ...
    };
```

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

The `crossOrigin` attribute is not supported.

V0058: The canPlayType function incorrectly returns a DOMString, not a CanPlayTypeEnum

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        CanPlayTypeEnum canPlayType(DOMString type);
        ...
    };
```

All document modes (All versions)

The `canPlayType` function incorrectly returns a `DOMString`, not a `CanPlayTypeEnum`.

V0059: The getStartDate function is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        Date getStartDate();
        ...
    };
```

All document modes (All versions)

The `getStartDate` function is not supported.

V0060: The mediaGroup attribute is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        attribute DOMString mediaGroup;
        ...
    };
```

All document modes (All versions)

The `mediaGroup` attribute is not supported.

V0061: The controller attribute is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        attribute MediaController? controller;
        ...
    };
```

All document modes (All versions)

The `controller` attribute is not supported.

V0062: The addTextTrack function is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        TextTrack addTextTrack(TextTrackKind kind, optional DOMString label = "",
        optional DOMString language = "");
    };
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `addTextTrack` function is not supported.

V0063: The defaultMuted attribute is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        attribute boolean defaultMuted;
        ...
    };
```

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

The `defaultMuted` attribute is not supported.

V0064: The audioTracks and textTracks attributes are not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        readonly attribute AudioTrackList audioTracks;
        ...
        readonly attribute TextTrackList textTracks;
        ...
    };
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `audioTracks` and `textTracks` attributes are not supported.

V0065: The HTMLMediaElement interface is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
    };
```

IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `HTMLMediaElement` interface is not supported.

V0066: The videoTracks attribute is not supported

The specification states:

```
4.7.10 Media elements
    ...
    interface HTMLMediaElement : HTMLElement {
        ...
        readonly attribute VideoTrackList videoTracks;
        ...
    };
```

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

The `videoTracks` attribute is not supported.

V0067: The CanPlayTypeEnum enum is not supported

The specification states:

```
4.7.10 Media elements
    Media elements (audio and video, in this specification) implement the following
    interface:

        enum CanPlayTypeEnum { "" /* empty string */, "maybe", "probably" };
```

All document modes (All versions)

The `CanPlayTypeEnum` enum is not supported.

2.1.36 [HTML5] Section 4.7.10.1 Error codes

V0068: The return value for the code attribute is a short, not an unsigned short

The specification states:

```
4.7.10.1 Error codes
    ...
    interface MediaError {
        ...
        readonly attribute unsigned short code;
    };
```

All document modes (All versions)

The return value for the `code` attribute is a `short`, not an `unsigned short`.

2.1.37 [HTML5] Section 4.7.10.2 Location of the media resource

V0069: The currentSrc value is not set to an absolute URL

The specification states:

```
4.7.10.2 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.
```

All document modes (All versions)

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

2.1.38 [HTML5] Section 4.7.10.5 Loading the media resource

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

The specification states:

```
4.7.10.5 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.
        ...
```

All document modes (All versions)

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:

```
4.7.10.5 Loading the media resource
    ...
    The resource fetch algorithm for a media element and a given absolute URL is as
    follows:
        ...
        3.  Optionally, run the following substeps. This is the expected behavior if the
            user agent intends to not attempt to fetch the resource until the user
            requests it explicitly (e.g. as a way to implement the preload attribute's
            none keyword).
            ...
            2.  Queue a task to fire a simple event named suspend at the element, using
                the DOM manipulation task source.
            ...
```

All document modes (All versions)

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

V0072: The network state is not set to NETWORK_EMPTY if src is empty

The specification states:

```
4.7.10.5 Loading the media resource
```

```
...
The resource fetch algorithm for a media element and a given absolute URL is as
follows:
    ...
    4.  Perform a potentially CORS-enabled fetch of the current media resource's
        absolute URL, with the mode being the state of the media element's
        crossorigin content attribute, the origin being the origin of the media
        element's Document, and the default origin behaviour set to taint.
        ...
        The networking task source tasks to process the data as it is being fetched
        must, when appropriate, include the relevant substeps from the following list:
            ...
            If the media data is corrupted
                Fatal errors in decoding the media data that occur after the user
                agent has established whether the current media resource is usable
                must cause the user agent to execute the following steps:
                    ...
                    4.  If the media element's readyState attribute has a value equal
                        to HAVE NOTHING, set the element's networkState attribute to
                        the NETWORK_EMPTY value, set the element's show poster flag
                        to true, and fire a simple event named emptied at the element.
```

All document modes (All versions)

The `networkState` is not set to `NETWORK_EMPTY` if `src` is empty.

V0073: The resource selection algorithm does not set the networkState correctly when load, play, or pause is called

The specification states:

```
4.7.10.5 Loading the media resource
    ...
    The resource selection algorithm for a media element is as follows. ...
        ...
        6.  If the media element has a src attribute, then let mode be attribute.

            Otherwise, if the media element does not have a src attribute but has a
            source element child, then let mode be children and let candidate be the
            first such source element child in tree order.

            Otherwise the media element has neither a src attribute nor a source element
            child: set the networkState to NETWORK EMPTY, and abort these steps; the
            synchronous section ends.
```

All document modes (All versions)

The resource selection algorithm does not set the `networkState` correctly when `load`, `play` or `pause` is called.

V0074: When the mode is attribute, the error events do not fire properly

The specification states:

```
4.7.10.5 Loading the media resource
    ...
    The resource selection algorithm for a media element is as follows. ...
        ...
```

```
            9.  If mode is attribute, then run these substeps:
                ...
                6.  Failed with attribute: Reaching this step indicates that the media
                    resource failed to load or that the given URL could not be resolved.
                    Queue a task to run the following steps, using the DOM manipulation task
                    source:
                    ...
                    5.  Fire a simple event named error at the media element.
```

All document modes (All versions)

When the `mode` is `attribute`, the error events do not fire properly.

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

The specification states:

```
4.7.10.5 Loading the media resource
    ...
    The resource selection algorithm for a media element is as follows. ...
        ...
        9.  If mode is attribute, then run these substeps:
            ...
            3.  If absolute URL was obtained successfully, set the currentSrc attribute
                to absolute URL.
```

All document modes (All versions)

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

2.1.39 [HTML5] Section 4.7.10.6 Offsets into the media resource

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

The specification states:

```
4.7.10.6 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. ...
```

All document modes (All versions)

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

2.1.40 [HTML5] Section 4.7.10.9 Seeking

V0077: The currentTime attribute updates asynchronously

The specification states:

```
4.7.10.9 Seeking
    ...
    When the user agent is required to seek to a particular new playback position in the
    media resource, optionally with the approximate-for-speed flag set, it means that the
    user agent must run the following steps.
        ...
        11.  Set the current playback position to the given new playback position.
             ...
             Note: The currentTime attribute does not get updated asynchronously, as it
             returns the official playback position, not the current playback position.
```

All document modes (All versions)

The `currentTime` attribute updates asynchronously.

2.1.41 [HTML5] Section 4.7.10.10.1 AudioTrackList and VideoTrackList objects

V0078: The AudioTrack attributes kind and language are not readonly

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    interface AudioTrack {
        ...
        readonly attribute DOMString kind;
        ...
        readonly attribute DOMString language;
        ...
    };
```

All document modes (All versions)

The `AudioTrack` attributes `kind` and `language` are not `readonly`.

V0079: The AudioTrack and AudioTrackList interfaces are not supported

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    interface AudioTrackList : EventTarget {
        ...
    };

    interface AudioTrack {
        ...
    };
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `AudioTrack` and `AudioTrackList` interfaces are not supported.

V0080: The VideoTrack and VideoTrackList interfaces are not supported

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    interface VideoTrackList : EventTarget {
        ...
    };

    interface VideoTrack {
        ...
    };
```

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

The `VideoTrack` and `VideoTrackList` interfaces are not supported.

V0081: The onremovetrack attribute of AudioTrackList is not supported

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    interface AudioTrackList : EventTarget {
        ...
        attribute EventHandler onremovetrack;
    };
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `onremovetrack` attribute of `AudioTrackList` is not supported.

V0082: At least one videoTrack in a videoTrackList must be selected

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    A VideoTrackList object represents a dynamic list of zero or more video tracks, of
    which zero or one can be selected at a time. ...
```

All document modes (All versions)

At least one `VideoTrack` in a `VideoTrackList` must be selected.

V0083: Media Fragments URI fragment identifiers are not supported

The specification states:

```
4.7.10.12.5 Text track API
    ...
    ... If the media resource is in a format that supports the Media Fragments URI
    fragment identifier syntax, the identifier returned for a particular track must be
```

```
the same identifier that would enable the track if used as the name of a track in the
track dimension of such a fragment identifier. ...
```

All document modes (All versions)

Media Fragments URI fragment identifiers are not supported.

V0084: AudioTrack.kind and VideoTrack.kind do not check if the category is appropriate for the media type

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    ... Categories must only be returned for AudioTrack objects if they are appropriate
    for audio, and must only be returned for VideoTrack objects if they are appropriate
    for video.
```

All document modes (All versions)

`AudioTrack.kind` and `VideoTrack.kind` do not check if the category is appropriate for the media type.

V0085: AudioTrack.language and VideoTrack.language return RFC-1766 language tags

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    The AudioTrack.language and VideoTrack.language attributes must return the BCP 47
    language tag of the language of the track, if it has one, or the empty string
    otherwise. ...
```

All document modes (All versions)

`AudioTrack.language` and `VideoTrack.language` return RFC-1766 language tags.

V0086: The resize event does not fire on a resize

The specification states:

```
4.7.10.10.1 AudioTrackList and VideoTrackList objects
    ...
    Whenever a track in a VideoTrackList that was previously not selected is selected,
    the user agent must queue a task to fire a simple event named change at the
    VideoTrackList object. This task must be queued before the task that fires the resize
    event, if any.
```

All document modes (All versions)

The `resize` event does not fire on a resize.

2.1.42 [HTML5] Section 4.7.10.10.2 Selecting specific audio and video tracks declaratively

V0087: Declarative selection of tracks is not supported

The specification states:

```
4.7.10.10.2 Selecting specific audio and video tracks declaratively

    The audioTracks and videoTracks attributes allow scripts to select which track should
    play, but it is also possible to select specific tracks declaratively, by specifying
    particular tracks in the fragment identifier of the URL of the media resource. The
    format of the fragment identifier depends on the MIME type of the media resource.
```

All document modes (All versions)

Declarative selection of tracks is not supported.

2.1.43 [HTML5] Section 4.7.10.11 Synchronizing multiple media elements

V0088: The MediaController interface is not supported

The specification states:

```
4.7.10.11.1 Introduction
    Each media element can have a MediaController. A MediaController is an object that
    coordinates the playback of multiple media elements, for instance so that a
    sign-language interpreter track can be overlaid on a video track, with the two being
    kept in sync.
```

All document modes (All versions)

The `MediaController` interface is not supported.

2.1.44 [HTML5] Section 4.7.10.12.1 Text track model

V0089: The change event is not fired when the text track mode changes

The specification states:

```
4.7.10.12.1 Text track model
    ...
    Whenever a text track that is in a media element's list of text tracks has its text
    track mode change value, the user agent must run the following steps for the media
    element:
        ...
        3.  Queue a task that runs the following substeps:
            ...
            2.  Fire a simple event named change at the media element's textTracks
                attribute's TextTrackList object.
```

All document modes (All versions)

The `change` event is not fired when the text track mode changes.

2.1.45 [HTML5] Section 4.7.10.12.2 Sourcing in-band text tracks

V0090: Ogg files are not supported

The specification states:

```
4.7.10.12.2 Sourcing in-band text tracks
    ...
    When a media resource contains data that the user agent recognises and supports as
    being equivalent to a text track, the user agent runs the steps to expose a
    media-resource-specific text track with the relevant data, as follows.
        ...
        4.  If the new text track's kind is metadata, then set the text track in-band
            metadata track dispatch type as follows, based on the type of the media
            resource:

                If the media resource is an Ogg file

                    The text track in-band metadata track dispatch type must be set to
                    the value of the Role header field.
```

All document modes (All versions)

Ogg files are not supported.

V0091: WebM files are not supported

The specification states:

```
4.7.10.12.2 Sourcing in-band text tracks
    ...
    When a media resource contains data that the user agent recognises and supports as
    being equivalent to a text track, the user agent runs the steps to expose a
    media-resource-specific text track with the relevant data, as follows.
        ...
        4.  If the new text track's kind is metadata, then set the text track in-band
            metadata track dispatch type as follows, based on the type of the media
            resource:
            ...
            If the media resource is a WebM file

                The text track in-band metadata track dispatch type must be set to the
                value of the CodecID element.
```

All document modes (All versions)

WebM files are not supported.

V0092: MPEG-4 metadata is not supported

The specification states:

```
4.7.10.12.2 Sourcing in-band text tracks
    ...
    When a media resource contains data that the user agent recognises and supports as
    being equivalent to a text track, the user agent runs the steps to expose a
    media-resource-specific text track with the relevant data, as follows.
        ...
        4.  If the new text track's kind is metadata, then set the text track in-band
            metadata track dispatch type as follows, based on the type of the media
            resource:
            ...
            If the media resource is an MPEG-4 file

                Let the first stsd box of the first stbl box of the first minf box of the
                first mdia box of the text track's trak box in the first moov box of the
                file be the stsd box, if any. ...
```

All document modes (All versions)

MPEG-4 metadata is not supported.

V0093: DASH metadata is not supported

The specification states:

```
4.7.10.12.2 Sourcing in-band text tracks
    ...
    When a media resource contains data that the user agent recognises and supports as
    being equivalent to a text track, the user agent runs the steps to expose a
    media-resource-specific text track with the relevant data, as follows.
        ...
        4.  If the new text track's kind is metadata, then set the text track in-band
            metadata track dispatch type as follows, based on the type of the media
            resource:
            ...
            If the media resource is a DASH media resource

                The text track in-band metadata track dispatch type must be set to the
                concatenation of the "AdaptationSet" element attributes and all child
                Role descriptors.
```

All document modes (All versions)

DASH metadata is not supported.

V0094: MPEG-2 support is limited to HLS and ID3 timed metadata

The specification states:

```
4.7.10.12.2 Sourcing in-band text tracks
    ...
    When a media resource contains data that the user agent recognises and supports as
    being equivalent to a text track, the user agent runs the steps to expose a
    media-resource-specific text track with the relevant data, as follows.
        ...
        4.  If the new text track's kind is metadata, then set the text track in-band
            metadata track dispatch type as follows, based on the type of the media
            resource:
            ...
            If the media resource is an MPEG-2 file
```

```
Let stream type be the value of the "stream_type" field describing the
text track's type in the file's program map section, interpreted as an
8-bit unsigned integer. ...
```

EdgeHTML Mode (All versions)

MPEG-2 support is limited to HLS only. MPEG-2 metadata support is limited to ID3 timed metadata.

V0095: MPEG-2 files are not supported

The specification states:

```
4.7.10.12.2 Sourcing in-band text tracks
    ...
    When a media resource contains data that the user agent recognises and supports as
    being equivalent to a text track, the user agent runs the steps to expose a
    media-resource-specific text track with the relevant data, as follows.
        ...
        4.  If the new text track's kind is metadata, then set the text track in-band
            metadata track dispatch type as follows, based on the type of the media
            resource:
            ...
            If the media resource is an MPEG-2 file

                Let stream type be the value of the "stream_type" field describing the
                text track's type in the file's program map section, interpreted as an
                8-bit unsigned integer. ...
```

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

MPEG-2 files are not supported.

2.1.46 [HTML5] Section 4.7.10.12.3 Sourcing out-of-band text tracks

V0096: No error event fires if the fetching algorithm fails

The specification states:

```
4.7.10.12.3 Sourcing out-of-band text tracks

    If the fetching algorithm fails for any reason (network error, the server returns an
    error code, a cross-origin check fails, etc), if URL is the empty string, or if the
    type of the resource is not a supported text track format, then run these steps:

        1.  Queue a task to first change the text track readiness state to failed to load
            and then fire a simple event named error at the track element.
```

All document modes (All versions)

No error event fires if the fetching algorithm fails.

V0097: The removetrack event does not fire when a text track is removed

The specification states:

```
4.7.10.12.3 Sourcing out-of-band text tracks
    ...
    When a track element's parent element changes and the old parent was a media element,
    then the user agent must remove the track element's corresponding text track from the
    media element's list of text tracks, and then queue a task to fire a trusted event
    with the name removetrack, that does not bubble and is not cancelable, and that uses
    the TrackEvent interface, with the track attribute initialized to the text track's
    TextTrack object, at the media element's textTracks attribute's TextTrackList object.
```

IE11 mode, IE10 mode, and EdgeHTML Mode (All versions)

The `removetrack` event does not fire when a text track is removed.

V0098: Text track selection is based on the default attribute only

The specification states:

```
4.7.10.12.3 Sourcing out-of-band text tracks
    ...
    When the steps above say to perform automatic text track selection for one or more
    text track kinds, it means to run the following steps:
        ...
        4.  If the user has expressed an interest in having a track from candidates
            enabled based on its text track kind, text track language, and text track
            label, then set its text track mode to showing.

            Otherwise, if there are any text tracks in candidates that correspond to
            track elements with a default attribute set whose text track mode is set to
            disabled, then set the text track mode of the first such track to showing.
```

IE11 mode, IE10 mode, and EdgeHTML Mode (All versions)

Text track selection is based on the `default` attribute only.

V0099: The crossorigin attribute is not supported

The specification states:

```
4.7.10.12.3 Sourcing out-of-band text tracks
    ...
    When a user agent is to start the track processing model for a text track and its
    track element, it must run the following algorithm.
        ...
        8.  If the track element's parent is a media element then let CORS mode be the
            state of the parent media element's crossorigin content attribute. Otherwise,
            let CORS mode be No CORS.
        ...
        10. If URL is not the empty string, perform a potentially CORS-enabled fetch of
            URL, with the mode being CORS mode, the origin being the origin of the track
            element's Document, and the default origin behaviour set to fail.

            The resource obtained in this fashion, if any, contains the text track data.
            If any data is obtained, it is by definition CORS-same-origin (cross-origin
            resources that are not suitably CORS-enabled do not get this far).
```

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

The `crossorigin` attribute is not supported.

2.1.47 [HTML5] Section 4.7.10.12.5 Text track API

V0100: The TextTrackList interface does not define the onchange or onremovetrack event handlers

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrackList : EventTarget {
        ...
        attribute EventHandler onchange;
        ...
        attribute EventHandler onremovetrack;
        ...
    };
```

All document modes (All versions)

The `TextTrackList` interface does not define the `onchange` or `onremovetrack` event handlers.

V0101: The TextTrackList interface does not define the getter correctly

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrackList : EventTarget {
        ...
        getter TextTrack (unsigned long index);
        ...
    };
```

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

The `TextTrackList` interface does not define the getter correctly:

getter TextTrack item(unsigned long index);

V0102: The TextTrackList interface does not define the onaddtrack event handler

The specification states:

```
4.7.10.12.5 Text track API

    interface TextTrackList : EventTarget {
        ...
        attribute EventHandler onaddtrack;
        ...
```

```
    };
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `TextTrackList` interface does not define the `onaddtrack` event handler.

V0103: The TextTrackList interface does not define the getTrackById function

The specification states:

```
4.7.10.12.5 Text track API

    interface TextTrackList : EventTarget {
        ...
        TextTrack? getTrackById(DOMString id);
        ...
    };
```

All document modes (All versions)

The `TextTrackList` interface does not define the `getTrackById` function.

V0104: The TextTrackMode and TextTrackKind enums are not defined

The specification states:

```
4.7.10.12.5 Text track API
    ...
    enum TextTrackMode { "disabled",  "hidden",  "showing" };
    enum TextTrackKind { "subtitles",  "captions",  "descriptions",  "chapters",
    "metadata" };
```

All document modes (All versions)

The `TextTrackMode` and `TextTrackKind` enums are not defined.

V0105: The kind attribute returns a DOMString, not a TextTrackKind

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrack : EventTarget {
        readonly attribute TextTrackKind kind;
        ...
    };
```

All document modes (All versions)

The `kind` attribute returns a `DOMString`, not a `TextTrackKind`.

V0106: The addCue and removeCue functions are not supported

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrack : EventTarget {
        ...
        void addCue(TextTrackCue cue);
        void removeCue(TextTrackCue cue);
        ...
    };
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `addCue` and `removeCue` functions are not supported.

V0107: The cues and activeCues attributes are not defined as nullable types

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrack : EventTarget {
        ...
        readonly attribute TextTrackCueList? cues;
        readonly attribute TextTrackCueList? activeCues;
        ...
    };
```

All document modes (All versions)

The `cues` and `activeCues` attributes are not defined as nullable types:

```
readonly attribute TextTrackCueList cues;

readonly attribute TextTrackCueList activeCues;
```

V0108: The id and inBandMetadataTrackDispatchType attributes are not supported

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrack : EventTarget {
        ...
        readonly attribute DOMString id;
        readonly attribute DOMString inBandMetadataTrackDispatchType;
        ...
    };
```

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

The `id` and `inBandMetadataTrackDispatchType` attributes are not supported.

V0109: The getCueById function does not return a nullable type

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrackCueList {
        ...
        TextTrackCue? getCueById(DOMString id);
    };
```

All document modes (All versions)

The `getCueById` function does not return a nullable type.

V0110: The track attribute does not return a nullable type

The specification states:

```
4.7.10.12.5 Text track API
    ...
    interface TextTrackCue : EventTarget {
        readonly attribute TextTrack? track;
        ...
    };
```

All document modes (All versions)

The `track` attribute does not return a nullable type:

```
readonly attribute TextTrack track;
```

V0111: The addontrack event handler is incorrectly defined as a nullable EventHandler

The specification states:

```
4.7.10.12.5 Text track API
    interface TextTrackList : EventTarget {
        ...
        attribute EventHandler onaddtrack;
        ...
    };
```

IE11 mode and EdgeHTML Mode (All versions)

The `onaddtrack` event handler is incorrectly defined as a nullable `EventHandler`.

2.1.48 [HTML5] Section 4.7.10.12.7 Event definitions

V0112: These TextTrackList, TextTrack, and TextTrackCue event handlers are not supported

The specification states:

```
4.7.10.12.7 Event definitions

    The following are the event handlers that (and their corresponding event handler
    event types) must be supported, as event handler IDL attributes, by all objects
    implementing the TextTrackList interface:

        Event handler | Event handler event type
        --------------------------------------
        onchange      | change
        onaddtrack    | addtrack
        onremovetrack | removetrack

    The following are the event handlers that (and their corresponding event handler
    event types) must be supported, as event handler IDL attributes, by all objects
    implementing the TextTrack interface:

        Event handler | Event handler event type
        --------------------------------------
        oncuechange   | cuechange

    The following are the event handlers that (and their corresponding event handler
    event types) must be supported, as event handler IDL attributes, by all objects
    implementing the TextTrackCue interface:

        Event handler | Event handler event type
        --------------------------------------
        onenter       | enter
        onexit        | exit
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

These `TextTrackList`, `TextTrack`, and `TextTrackCue` event handlers are not supported.

V0113: The onaddtrack event handler is not supported

The specification states:

```
4.7.10.12.7 Event definitions

    The following are the event handlers that (and their corresponding event handler
    event types) must be supported, as event handler IDL attributes, by all objects
    implementing the TextTrackList interface:

        Event handler | Event handler event type
        --------------------------------------
        onchange      | change
        onaddtrack    | addtrack
        onremovetrack | removetrack
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `onaddtrack` event handler is not supported.

2.1.49 [HTML5] Section 4.7.10.14 Time ranges

V0114: The start and end methods throw the wrong exception

The specification states:

```
4.7.10.14 Time ranges

    The start(index) method must return the position of the start of the index'th range
    represented by the object, in seconds measured from the start of the timeline that
    the object covers.

    The end(index) method must return the position of the end of the index'th range
    represented by the object, in seconds measured from the start of the timeline that
    the object covers.

    These methods must throw IndexSizeError exceptions if called with an index argument
    greater than or equal to the number of ranges represented by the object.
```

All document modes (All versions)

If called with an index argument greater than or equal to the number of ranges represented by the object, `start` and `end` throw an invalid argument exception, not an `IndexSizeError` exception.

2.1.50 [HTML5] Section 4.7.10.15 Event definitions

V0115: The TrackEventInit dictionary is not supported

The specification states:

```
4.7.10.15 Event definitions
    ...
    dictionary TrackEventInit : EventInit {
        (VideoTrack or AudioTrack or TextTrack) track;
    };
```

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

The `TrackEventInit` dictionary is not supported.

2.1.51 [HTML5] Section 4.7.11 The map element

V0116: The areas collection is returned as an HTMLAreasCollection, not an HTMLCollection

The specification states:

```
4.7.11 The map element
    ...
    interface HTMLMapElement : HTMLElement {
        ...
        readonly attribute HTMLCollection areas;
        ...
     };
```

All document modes (All versions)

The `areas` collection is returned as an `HTMLAreasCollection`, not an `HTMLCollection`.

V0117: The images collection is not supported

The specification states:

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

All document modes (All versions)

The `images` collection is not supported.

2.1.52 [HTML5] Section 4.7.12 The area element

V0118: The relList attribute is not supported

The specification states:

```
4.7.12 The area element
    ...
    interface HTMLAreaElement : HTMLElement {
        ...
        readonly attribute DOMTokenList relList;
        ...
    };
```

All document modes (All versions)

The `relList` attribute is not supported.

V0119: The download attribute is not supported

The specification states:

```
4.7.12 The area element
    ...
    interface HTMLAreaElement : HTMLElement {
        ...
        attribute DOMString download;
        ...
    };
```

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

The `download` attribute is not supported.

V0120: The hreflang attribute is not supported

The specification states:

```
4.7.12 The area element
    ...
    interface HTMLAreaElement : HTMLElement {
        ...
        attribute DOMString hreflang;
        ...
    };
```

All document modes (All versions)

The `hreflang` attribute is not supported.

V0121: The type attribute is not supported

The specification states:

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

All document modes (All versions)

The `type` attribute is not supported.

V0122: The URLUtils interface is not implemented for the HTMLAreaElement interface

The specification states:

```
4.7.12 The area element
    ...
    interface HTMLAreaElement : HTMLElement {
        ...
    };

    HTMLAreaElement implements URLUtils;
```

All document modes (All versions)

The `URLUtils` interface is not implemented for the `HTMLAreaElement` interface.

However, some `URLUtils` attributes are implemented on instances of `HTMLAreaElement`. They are:

`href`

`protocol`

`host`

`hostname`

`port`

`pathname`

`search`

`hash`

These are not implemented:

`username`

`password`

`searchParams`

`origin`

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

The specification states:

```
4.7.12 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
```

All document modes (All versions)

The `default` keyword value is not supported.

V0124: For the circle state the last integer can be negative

The specification states:

```
4.7.12 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.
```

All document modes (All versions)

For the circle state the last integer can be negative; the radius is the absolute value of the integer.

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

The specification states:

```
4.7.12 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.
```

All document modes (All versions)

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:

```
4.7.12 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.
```

All document modes (All versions)

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

2.1.53 [HTML5] Section 4.7.13.2 Processing model

V0127: When images do not load, a valid image map will still be applied to the missing image

The specification states:

```
4.7.13.2 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.
```

All document modes (All versions)

When images do not load, a valid image map will still be applied to the missing image and not represented in a way associated with the text.

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

The specification states:

```
4.7.13.2 Processing model

    If an img element or an object element representing an image has a usemap attribute
    specified, user agents must process it as follows:

        1.  First, rules for parsing a hash-name reference to a map element must be
            followed. This will return either an element (the map) or null.
```

All document modes (All versions)

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

2.1.54 [HTML5] Section 4.7.14 MathML

V0129: The math element is not supported

The specification states:

```
4.7.14 MathML

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

All document modes (All versions)

The `math` element is not supported.

2.1.55 [HTML5] Section 4.8.3 Downloading resources

V0130: The download attribute is not supported

The specification states:

```
4.8.3 Downloading resources

    In some cases, resources are intended for later use rather than immediate viewing. To
    indicate that a resource is intended to be downloaded for use later, rather than
    immediately used, the download attribute can be specified on the a or area element
    that creates the hyperlink to that resource.
```

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

The `download` attribute is not supported.

2.1.56 [HTML5] Section 4.8.4.9 Link type "prefetch"

V0131: The prefetch link type is not supported for the a and area elements

The specification states:

```
4.8.4.9 Link type "prefetch"

    The prefetch keyword may be used with link, a, and area elements. This keyword
    creates an external resource link.

    The prefetch keyword indicates that preemptively fetching and caching the specified
    resource is likely to be beneficial, as it is highly likely that the user will
    require this resource.
```

All document modes (All versions)

The `prefetch` link type is not supported for the `a` and `area` elements.

2.1.57 [HTML5] Section 4.9.1 The table element

V0132: The insertRow function incorrectly inserts new rows

The specification states:

```
4.9.1 The table element
    ...
    The behavior of the insertRow(index) method depends on the state of the table. When
    it is called, the method must act as required by the first item in the following list
    of conditions that describes the state of the table and the index argument:

        If index is less than −1 or greater than the number of elements in rows
        collection:

            The method must throw an IndexSizeError exception.

        If the rows collection has zero elements in it, and the table has no tbody
        elements in it:

            The method must create a tbody element, then create a tr element, then append
            the tr element to the tbody element, then append the tbody element to the
            table element, and finally return the tr element.

        If the rows collection has zero elements in it:

            The method must create a tr element, append it to the last tbody element in
            the table, and return the tr element.

        If index is −1 or equal to the number of items in rows collection:

            The method must create a tr element, and append it to the parent of the last
            tr element in the rows collection. Then, the newly created tr element must be
            returned.

        Otherwise:

            The method must create a tr element, insert it immediately before the indexth
            tr element in the rows collection, in the same parent, and finally must
            return the newly created tr element.
```

All document modes (All versions)

The `insertRow` function incorrectly inserts new rows:

- When the `<table>` is empty, a `<tbody>` will be created.

- When the `<table>` contains only a `<thead>`, insertRow will add a new `<tr>` to the `<thead>` - a `<tbody>` is not added.

- When the `<table>` contains a `<tfoot>`, insertRow will add a new `<tr>` to the `<tfoot>` - a `<tbody>` is not added.

2.1.58 [HTML5] 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

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

All document modes (All versions)

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

V0134: The deleteRow function deletes the last row if index is -1

The specification states:

```
4.9.5 The tbody element
    ...
    The deleteRow(index) method must remove the index'th element in the rows collection
    from its parent. If index is less than zero or greater than or equal to the number of
    elements in the rows collection, the method must instead throw an IndexSizeError
    exception.
```

All document modes (All versions)

The `deleteRow` function deletes the last row if `index` is -1.

2.1.59 [HTML5] 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
    ...
    interface HTMLTableRowElement : HTMLElement {
        ...
```

```
        void deleteCell(long index);
    };
```

All document modes (All versions)

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

V0136: If index is -1, the deleteCell method deletes the last cell of the row

The specification states:

```
4.9.8 The tr element
    ...
    The deleteCell(index) method must remove the index'th element in the cells collection
    from its parent. If index is less than zero or greater than or equal to the number of
    elements in the cells collection, the method must instead throw an IndexSizeError
    exception.
```

All document modes (All versions)

If index is -1, the `deleteCell` method deletes the last cell of the row.

2.1.60 [HTML5] Section 4.9.10 The th element

V0137: The abbr attribute is abstracted to the base class HTMLTableCellElement

The specification states:

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

All document modes (All versions)

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

2.1.61 [HTML5] Section 4.9.11 Attributes common to td and th elements

V0138: The headers attribute is not supported

The specification states:

```
4.9.11 Attributes common to td and th elements
    ...
    interface HTMLTableCellElement : HTMLElement {
        ...
        [PutForwards=value] readonly attribute DOMSettableTokenList headers;
```

```
        ...
    };
```

IE11 mode, IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (Internet Explorer 11, Internet Explorer 10, Internet Explorer 9, and Internet Explorer 8)

The `headers` attribute is not supported.

2.1.62 [HTML5] Section 4.9.12.2 Forming relationships between data cells and header cells

V0139: The headers attribute is not supported

The specification states:

```
4.9.12.2 Forming relationships between data cells and header cells
```

IE11 mode, IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (Internet Explorer 11, Internet Explorer 10, Internet Explorer 9, and Internet Explorer 8)

The `headers` attribute is not supported.

2.1.63 [HTML5] Section 4.10.3 The form element

V0140: The novalidate attribute is not supported

The specification states:

```
4.10.3 The form element
    ...
    [OverrideBuiltins]
    interface HTMLFormElement : HTMLElement {
        ...
           attribute boolean noValidate;
        ...
    };
```

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

The `novalidate` attribute is not supported.

V0141: The elements collection is of type HTMLCollection, not HTMLFormControlsCollection

The specification states:

```
4.10.3 The form element
    ...
    [OverrideBuiltins]
    interface HTMLFormElement : HTMLElement {
        ...
```

```
            readonly attribute HTMLFormControlsCollection elements;
        ...
    };
```

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

The `elements` attribute is of type `HTMLCollection`, not `HTMLFormControlsCollection`.

2.1.64 [HTML5] Section 4.10.4 The label element

V0142: The control attribute is not supported

The specification states:

```
4.10.4 The label element
    ...
    interface HTMLLabelElement : HTMLElement {
        ...
        readonly attribute HTMLElement? control;
    };
```

All document modes (All versions)

The `control` attribute is not supported.

2.1.65 [HTML5] Section 4.10.5 The input element

V0143: The valueAsDate attribute is not supported

The specification states:

```
4.10.5 The input element
    ...
    interface HTMLInputElement : HTMLElement {
        ...
        attribute Date? valueAsDate;
        ...
    };
```

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

The `valueAsDate` attribute is not supported.

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:
    ...
    4. Invoke the value sanitization algorithm, if one is defined for the type
       attribute's new state.
```

All document modes (All versions)

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

The specification states:

```
4.10.5 The input element
    ...
    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,
        optionalSelectionMode selectionMode = "preserve");
        void setSelectionRange(unsigned long start, unsigned long end, optional DOMString
        direction);
    };
```

All document modes (All versions)

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

V0146: The dirName attribute is not supported

The specification states:

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

All document modes (All versions)

The `dirName` attribute is not supported.

V0147: The labels attribute is not supported

The specification states:

```
4.10.5 The input element
    ...
    interface HTMLInputElement : HTMLElement {
        ...
        readonly attribute NodeList labels;
        ...
    };
```

All document modes (All versions)

The `labels` attribute is not supported.

V0148: The minLength attribute is not supported

The specification states:

```
4.10.5 The input element
    ...
    interface HTMLInputElement : HTMLElement {
        ...
        attribute long minLength;
        ...
    };
```

All document modes (All versions)

The `minLength` attribute is not supported.

V0149: The setRangeText functions are not supported

The specification states:

```
4.10.5 The input element
    ...
    interface HTMLInputElement : HTMLElement {
        ...
        void setRangeText(DOMString replacement);
        void setRangeText(DOMString replacement, unsigned long start, unsigned long end,
        optional SelectionMode selectionMode = "preserve");
        ...
    };
```

All document modes (All versions)

The `setRangeText` functions are not supported.

2.1.66 [HTML5] Section 4.10.5.1.1 Hidden state (type=hidden)

V0150: The files attribute incorrectly return undefined, not null.

The specification states:

```
4.10.5.1.1 Hidden state (type=hidden)
```

```
...
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.
```

All document modes (All versions)

The `files` attribute incorrectly returns `undefined`, not `null`;

2.1.67 [HTML5] Section 4.10.5.1.4 URL state (type=url)

V0151: The value sanitization implementation does not properly strip leading and trailing whitespace from a URI

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.
```

All document modes (All versions)

The value sanitization implementation does not strip leading and trailing whitespace from a URL.

2.1.68 [HTML5] Section 4.10.5.1.5 E-mail state (type=email)

V0152: The value sanitization algorithm is not run when the multiple attribute is removed

The specification states:

```
4.10.5.1.5 E-mail state (type=email)
    ...
    How the E-mail state operates depends on whether the multiple attribute is specified
    or not.

        When the multiple attribute is not specified on the element
            ...
            When the multiple attribute is removed, the user agent must run the value
            sanitization algorithm.
```

All document modes (All versions)

The value sanitization algorithm is not run when the `multiple` attribute is removed.

V0153: Email addresses are not properly validated when the multiple attribute is specified

The specification states:

```
4.10.5.1.5 E-mail state (type=email)
    ...
```

```
How the E-mail state operates depends on whether the multiple attribute is specified
or not.
    ...
    When the multiple attribute is specified on the element
        ...
        Constraint validation: While the user interface describes a situation where
        an individual value contains a "," (U+002C) or is representing input that the
        user agent cannot convert to punycode, the control is suffering from bad
        input.

        Whenever the user changes the element's values, the user agent must run the
        following steps:

            Let latest values be a copy of the element's values.

            Strip leading and trailing whitespace from each value in latest values.

            Let the element's value be the result of concatenating all the values in
            latest values, separating each value from the next by a single ","
            (U+002C) character, maintaining the list's order.

        The value attribute, if specified, must have a value that is a valid e-mail
        address list.

        The value sanitization algorithm is as follows:

            Split on commas the element's value, strip leading and trailing
            whitespace from each resulting token, if any, and let the element's
            values be the (possibly empty) resulting list of (possibly empty) tokens,
            maintaining the original order.

            Let the element's value be the result of concatenating the element's
            values, separating each value from the next by a single "," (U+002C)
            character, maintaining the list's order.

        When the multiple attribute is set, the user agent must run the value
        sanitization algorithm.

        Constraint validation: While the value of the element is not a valid e-mail
        address list, the element is suffering from a type mismatch.
```

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

Email addresses are not properly validated when the `multiple` attribute is specified and there are multiple email addresses, each separated from the next by a single "," (U+002C) character.

2.1.69 [HTML5] Section 4.10.5.1.7 Date state (type=date)

V0154: The Date state is not supported

The specification states:

```
4.10.5.1.7 Date state (type=date)
    ...
    When an input element's type attribute is in the Date state, the rules in this
    section apply.
```

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

The `Date` state is not supported.

2.1.70 [HTML5] Section 4.10.5.1.8 Time state (type=time)

V0155: The Time state is not supported

The specification states:

```
4.10.5.1.8 Time state (type=time)
    ...
    When an input element's type attribute is in the Time state, the rules in this
    section apply.
```

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

The `Time` state is not supported.

2.1.71 [HTML5] Section 4.10.5.1.9 Number state (type=number)

V0156: Number with a leading whitespace results in empty value

The specification states:

```
4.10.5.1.9 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.
```

All document modes (All versions)

The rules for parsing floating-point number values specify that white space should be skipped. This is not done, however, and the presence of white space is treated as invalid input.

2.1.72 [HTML5] Section 4.10.5.1.10 Range state (type=range)

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

The specification states:

```
4.10.5.1.10 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.
```

All document modes (All versions)

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:

```
4.10.5.1.10 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.
```

All document modes (All versions)

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

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

The specification states:

```
4.10.5.1.10 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).
```

All document modes (All versions)

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

2.1.73 [HTML5] Section 4.10.5.1.11 Color state (type=color)

V0160: The Color state is not supported

The specification states:

```
4.10.5.1.11 Color state (type=color)
    ...
    When an input element's type attribute is in the Color state, the rules in this
    section apply.
```

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

The `Color` state is not supported.

2.1.74 [HTML5] Section 4.10.5.1.12 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:

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

All document modes (All versions)

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:

```
4.10.5.1.12 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.
```

All document modes (All versions)

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

V0163: The validityState valueMissing returns false when it is required and checkedness is false

The specification states:

```
4.10.5.1.12 Checkbox state (type=checkbox)
    ...
    Constraint validation: If the element is required and its checkedness is false, then
    the element is suffering from being missing.
```

All document modes (All versions)

The validityState `valueMissing` returns false when it is required and checkedness is false.

2.1.75 [HTML5] Section 4.10.5.1.13 Radio Button state (type=radio)

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

The specification states:

```
4.10.5.1.13 Radio Button state (type=radio)
    ...
```

```
Bookkeeping details
    ...
    The input and change events apply
```

All document modes (All versions)

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

V0165: The preventDefault function does not properly cancel events

The specification states:

```
4.10.5.1.13 Radio Button state (type=radio)
    ...
    If the element is mutable, then: The pre-click activation steps consist of setting
    the element's checkedness to true. The canceled activation steps consist of setting
    the element's checkedness to false. 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.
```

All document modes (All versions)

The `preventDefault` function does not cancel events.

V0166: The name attribute for radio grouping is not compared in a compatibility caseless manner for all Unicode ranges

The specification states:

```
4.10.5.1.13 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.
```

All document modes (All versions)

The `name` attribute for radio grouping is not compared in a compatibility caseless manner for all Unicode ranges. The comparison uses ASCII comparison.

V0167: The valueMissing attribute returns false when an element is required and checkedness is false

The specification states:

```
4.10.5.1.13 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.
```

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

The `valueMissing` attribute of the `ValidityState` interface returns false when an input element is required and when checkedness is false for all input elements.

V0168: The checkedness state is incorrect for radio buttons in a group

The specification states:

```
4.10.5.1.13 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).
```

All document modes (All versions)

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

2.1.76 [HTML5] Section 4.10.5.1.14 File Upload state (type=file)

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

The specification states:

```
4.10.5.1.14 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).
```

All document modes (All versions)

The input type=`file` does not properly secure the selected file and obscure the local file location. It obscures the file when it is submitted to the server.

2.1.77 [HTML5] Section 4.10.5.1.17 Reset Button state (type=reset)

V0170: The input reset type does not reset all form controls

The specification states:

```
4.10.5.1.17 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.
```

All document modes (All versions)

The input type=`reset` does not reset form controls linked using the `form` attribute.

2.1.78 [HTML5] Section 4.10.6 The button element

V0171: The labels attribute is not supported

The specification states:

```
4.10.6 The button element
    ...
    interface HTMLButtonElement : HTMLElement {
        ...
        readonly attribute NodeList labels;
    };
```

All document modes (All versions)

The `labels` attribute is not supported.

2.1.79 [HTML5] Section 4.10.7 The select element

V0172: The options collection returns an HTMLSelectElement, not an HTMLOptionsCollection

The specification states:

```
4.10.7 The select element
    ...
    interface HTMLSelectElement : HTMLElement {
        ...
        readonly attribute HTMLOptionsCollection options;
        ...
    };
```

IE11 mode, IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (Internet Explorer 11, Internet Explorer 10, Internet Explorer 9, and Internet Explorer 8)

The `options` collection returns an `HTMLSelectElement`, not an `HTMLOptionsCollection`.

V0173: The labels attribute is not supported

The specification states:

```
4.10.7 The select element
```

```
...
interface HTMLSelectElement : HTMLElement {
    ...
    readonly attribute NodeList labels;
};
```

All document modes (All versions)

The `labels` attribute is not supported.

V0174: The selectedOptions collection is not supported

The specification states:

```
4.10.7 The select element
    ...
    interface HTMLSelectElement : HTMLElement {
        ...
        readonly attribute HTMLCollection selectedOptions;
        ...
    };
```

IE11 mode, IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (Internet Explorer 11, Internet Explorer 10, Internet Explorer 9, and Internet Explorer 8)

The `selectedOptions` collection is not supported.

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

The specification states:

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

All document modes (All versions)

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

V0176: The namedItem function does not select options based on the name attribute

The specification states:

```
4.10.7 The select element
    ...
    The namedItem(name) method must return the value returned by the method of the same
    name on the options collection, when invoked with the same argument.
```

IE11 mode, IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (Internet Explorer 11, Internet Explorer 10, Internet Explorer 9, and Internet Explorer 8)

The `namedItem` function does not select options based on the `name` attribute.

2.1.80 [HTML5] Section 4.10.10 The option element

V0177: The text attribute incorrectly includes text from SVG script elements

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 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.
```

All document modes (All versions)

When there is a nested SVG script element, the text of that element is included in the `DomString` returned by the `text` attribute of the `option` element.

2.1.81 [HTML5] Section 4.10.11 The textarea element

V0178: The labels attribute is not supported

The specification states:

```
4.10.11 The textarea element
    ...
    interface HTMLTextAreaElement : HTMLElement {
        ...
        readonly attribute NodeList labels;
        ...
    };
```

All document modes (All versions)

The `labels` attribute is not supported.

V0179: The textLength attribute is not supported

The specification states:

```
4.10.11 The textarea element
    ...
    interface HTMLTextAreaElement : HTMLElement {
        ...
        readonly attribute unsigned long textLength;
        ...
```

```
    };
```

All document modes (All versions)

The `textLength` attribute is not supported.

V0180: The autocomplete attribute is not supported

The specification states:

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

All document modes (All versions)

The `autocomplete` attribute is not supported.

V0181: The dirName attribute is not supported

The specification states:

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

All document modes (All versions)

The `dirName` attribute is not supported.

V0182: The minLength attribute is not supported

The specification states:

```
4.10.11 The textarea element
    ...
    interface HTMLTextAreaElement : HTMLElement {
        ...
        attribute long minLength;
        ...
    };
```

All document modes (All versions)

The `minLength` attribute is not supported.

2.1.82 [HTML5] Section 4.10.12 The keygen element

V0183: The keygen element is not supported

The specification states:

```
4.10.12 The keygen element
    ...
    The keygen element represents a key pair generator control. When the control's form
    is submitted, the private key is stored in the local keystore, and the public key is
    packaged and sent to the server.
```

All document modes (All versions)

The `keygen` element is not supported.

2.1.83 [HTML5] Section 4.10.13 The output element

V0184: The output element is not supported

The specification states:

```
4.10.13 The output element
    ...
    The output element represents the result of a calculation or user action.
```

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

The `output` element is not supported.

2.1.84 [HTML5] Section 4.10.14 The progress element

V0185: The labels attribute is not supported

The specification states:

```
4.10.14 The progress element
    ...
    interface HTMLProgressElement : HTMLElement {
        ...
        readonly attribute NodeList labels;
    };
```

All document modes (All versions)

The `labels` attribute is not supported.

2.1.85 [HTML5] Section 4.10.15 The meter element

V0186: The meter element is not supported

The specification states:

```
4.10.15 The meter element
    ...
    The meter element represents a scalar measurement within a known range, or a
    fractional value; for example disk usage, the relevance of a query result, or the
    fraction of a voting population to have selected a particular candidate.
```

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

The `meter` element is not supported.

2.1.86 [HTML5] Section 4.10.16 The fieldset element

V0187: The name attribute is not supported

The specification states:

```
4.10.16 The fieldset element
    ...
    interface HTMLFieldSetElement : HTMLElement {
        ...
        attribute DOMString name;
        ...
    };
```

All document modes (All versions)

The `name` attribute is not supported.

V0188: The type attribute is not supported

The specification states:

```
4.10.16 The fieldset element
    ...
    interface HTMLFieldSetElement : HTMLElement {
        ...
        readonly attribute DOMString type;
        ...
    };
```

All document modes (All versions)

The `type` attribute is not supported.

V0189: The elements collection is not supported

The specification states:

```
4.10.16 The fieldset element
    ...
    interface HTMLFieldSetElement : HTMLElement {
        ...
        readonly attribute HTMLFormControlsCollection elements;
        ...
    };
```

All document modes (All versions)

The `elements` collection is not supported.

2.1.87 [HTML5] Section 4.10.18.3 Association of controls and forms

V0190: The form attribute does not associate a form-associateable element with a form

The specification states:

```
4.10.18.3 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.
```

All document modes (All versions)

The `form` attribute does not associate a form-associateable element with a form.

2.1.88 [HTML5] Section 4.10.19.2 Submitting element directionality: the dirname attribute

V0191: The dirname attribute is not supported

The specification states:

```
4.10.19.2 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.
```

All document modes (All versions)

The `dirname` attribute is not supported.

2.1.89 [HTML5] Section 4.10.19.4 Setting minimum input length requirements: the minlength attribute

V0192: The minlength attribute is not supported

The specification states:

```
4.10.19.4 Setting minimum input length requirements: the minlength attribute

    A form control minlength attribute, controlled by a dirty value flag, declares a
    lower bound on the number of characters a user can input.
```

All document modes (All versions)

The `minlength` attribute is not supported.

2.1.90 [HTML5] Section 4.10.19.7 Autofocusing a form control: the autofocus attribute

V0193: The autofocus attribute does not always correctly focus on the correct control

The specification states:

```
4.10.19.7 Autofocusing a form control: the autofocus attribute

    The autofocus content attribute allows the author to indicate that a control is to be
    focused as soon as the page is loaded, allowing the user to just start typing without
    having to manually focus the main control.
```

IE11 mode, IE10 mode, IE9 mode, and IE8 mode (Internet Explorer 11, Internet Explorer 10, Internet Explorer 9, and Internet Explorer 8)

The `autofocus` attribute does not always correctly focus on the correct control.

2.1.91 [HTML5] Section 4.10.20 APIs for the text field selections

V0194: The selectionDirection attribute is not supported

The specification states:

```
4.10.20 APIs for the text field selections

    The input and textarea elements define the following members in their DOM interfaces
    for handling their selection:
        ...
        attribute DOMString selectionDirection;
```

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

The `selectionDirection` attribute is not supported.

V0195: The setRangeText functions are not supported

The specification states:

```
4.10.20 APIs for the text field selections

    The input and textarea elements define the following members in their DOM interfaces
    for handling their selection:
        ...
        void setRangeText(DOMString replacement);
        void setRangeText(DOMString replacement, unsigned long start, unsigned long end,
        optional SelectionMode selectionMode = "preserve");
```

All document modes (All versions)

The `setRangeText` functions are not supported.

2.1.92 [HTML5] Section 4.10.21.1 Definitions

V0196: The setCustomValidity function is not supported

The specification states:

```
4.10.21.1 Definitions
    ...
    An element can have a custom validity error message defined. Initially, an element
    must have its custom validity error message set to the empty string. When its value
    is not the empty string, the element is suffering from a custom error. It can be set
    using the setCustomValidity() method. ...
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `setCustomValidity` function is not supported.

2.1.93 [HTML5] Section 4.10.21.2 Constraint validation

V0197: Contraint validation does not return a list of invalid controls

The specification states:

```
4.10.21.2 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.
```

All document modes (All versions)

A negative result is returned, but not a list of elements for the unhandled invalid controls.

V0198: Constraint validation is not supported

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.
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

Constraint validation is not supported.

2.1.94 [HTML5] Section 4.10.21.3 The constraint validation API

V0199: The tooShort attribute is not supported

The specification states:

```
4.10.21.3 The constraint validation API
    ...
    interface ValidityState {
        ...
        readonly attribute boolean tooShort;
        ...
    };
```

All document modes (All versions)

The `tooShort` attribute is not supported.

V0200: The validity attribute is not supported

The specification states:

```
4.10.21.3 The constraint validation API
    ...
    The validity attribute must return a ValidityState object that represents the
    validity states of the element. This object is live, and the same object must be
    returned each time the element's validity attribute is retrieved.
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `validity` attribute is not supported.

V0201: The setCustomValidity function is not supported

The specification states:

```
4.10.21.3 The constraint validation API
    ...
    The setCustomValidity(message), when invoked, must set the custom validity error
    message to the value of the given message argument.
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `setCustomValidity` function is not supported.

V0202: The tooLong attribute returns true when the maxlength value is not set

The specification states:

```
4.10.21.3 The constraint validation API
    ...
    A ValidityState object has the following attributes. On getting, they must return
    true if the corresponding condition given in the following list is true, and false
    otherwise.
        ...
        tooLong
            The control is suffering from being too long.
```

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

The `tooLong` attribute returns true when the `maxlength` value is not set.

V0203: The badInput attribute is not supported

The specification states:

```
4.10.21.3 The constraint validation API
    ...
    interface ValidityState {
        ...
        readonly attribute boolean badInput;
        ...
    };
```

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

The `badInput` attribute is not supported.

2.1.95 [HTML5] Section 4.10.22.5 Selecting a form submission encoding

V0204: UTF-8 is used in form submission instead of the encoding in the accept-charset attribute

The specification states:

```
4.10.22.5 Selecting a form submission encoding

    If the user agent is to pick an encoding for a form, optionally with an allow
    non-ASCII-compatible encodings flag set, it must run the following substeps:

        7.  ...
            Return the first encoding in candidate encodings that can encode max
            characters in the names and values of the entries in the form data set.
```

All document modes (All versions)

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

2.1.96 [HTML5] Section 4.10.22.6 URL-encoded form data

V0205: URL-encoded form data is encoded in UTF-8 regardless of what is in accept-charset

The specification states:

```
4.10.22.6 URL-encoded form data
    ...
    The application/x-www-form-urlencoded encoding algorithm is as follows:
        ...
        2.  If the form element has an accept-charset attribute, let the selected
            character encoding be the result of picking an encoding for the form.

            Otherwise, if the form element has no accept-charset attribute, but the
            document's character encoding is an ASCII-compatible character encoding, then
            that is the selected character encoding.

            Otherwise, let the selected character encoding be UTF-8.
```

All document modes (All versions)

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:

```
4.10.22.6 URL-encoded form data
    ...
    The application/x-www-form-urlencoded encoding algorithm is as follows:
        ...
        4.  For each entry in the form data set, perform these substeps:
            ...
            2.  If the entry's type is "file", replace its value with the file's name
                only.
```

All document modes (All versions)

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

2.1.97 [HTML5] Section 4.10.22.7 Multipart form data

V0207: Forms are always encoded as UTF-8

The specification states:

```
4.10.22.7 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.
```

All document modes (All versions)

Forms are always encoded as UTF-8.

2.1.98 [HTML5] Section 4.10.22.8 Plain text form data

V0208: The submitted data set includes not only the filename but also the full path of the file

The specification states:

```
4.10.22.8 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.
```

All document modes (All versions)

The submitted data set includes not only the filename but also the full path of the file.

2.1.99 [HTML5] Section 4.11.1 The script element

V0209: The crossOrigin attribute is not supported

The specification states:

```
4.11.1 The script element
    ...
    interface HTMLScriptElement : HTMLElement {
        ...
        attribute DOMString crossOrigin;
        ...
    };
```

All document modes (All versions)

The `crossOrigin` attribute is not supported.

V0210: The async attribute is not supported

The specification states:

```
4.11.1 The script element
    ...
    interface HTMLScriptElement : HTMLElement {
        ...
        attribute boolean async;
        ...
    };
```

IE9 mode and IE8 mode (All versions)

The `async` attribute is not supported.

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

The specification states:

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

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

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

                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".
```

All document modes (All versions)

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 set to a value other than window, the script is still executed

The specification states:

```
4.11.1 The script element
    ...
    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.
```

All document modes (All versions)

If either the `for` or the `event` attribute is set to a value other than `window` or `onload` respectively, the script is still executed.

V0213: The beforescriptexecute and afterscriptexecute events were removed from the HTML5 standard, but this section still references them

The specification states:

```
4.11.1 The script element
    ...
    When the user agent is required to execute a script block, it must run the following
    steps:
        ...
        2.  Jump to the appropriate set of steps from the list below:
                ...
                If the load was successful

                    Executing the script block must consist of running the following
                    steps. ...
                        ...
                        2.  Fire a simple event named beforescriptexecute that bubbles
                            and is cancelable at the script element.
                        ...
                        6.  Fire a simple event named afterscriptexecute that bubbles
                            (but is not cancelable) at the script element.
```

All document modes (All versions)

The `beforescriptexecute` and `afterscriptexecute` events were removed from the HTML5 standard, but this section still references them.

2.1.100 [HTML5] Section 4.11.1.1 Scripting languages

V0214: Not all MIME types are recognized

The specification states:

```
4.11.1.1 Scripting languages
    ...
    The following lists the MIME type strings that user agents must recognize, and the
    languages to which they refer:
        "application/ecmascript"
        ...
        "text/javascript1.0"
        ...
        "text/javascript1.4"
        "text/javascript1.5"
```

```
...
"text/x-ecmascript"
```

All document modes (All versions)

The following MIME types are not recognized:

InlineCode|application/x-ecmascript]

`text/javascript1.0`

`text/javascript1.4`

`text/javascript1.5`

`text/x-ecmascript`

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

The specification states:

```
4.11.1.1 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.
```

All document modes (All versions)

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.

2.1.101 [HTML5] Section 4.11.3 The template element

V0216: The template element is not supported

The specification states:

```
4.11.3 The template element
    ...
    The template element is used to declare fragments of HTML that can be cloned and
    inserted in the document by script.
```

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

The `template` element is not supported.

2.1.102 [HTML5] Section 4.11.4 The canvas element

V0217: The toBlob function is not supported

The specification states:

```
4.11.4 The canvas element
    ...
    void toBlob(FileCallback?  callback, optional DOMString type, any... arguments);
};
```

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

The `toBlob` function is not supported.

V0218: The null character \0 is not properly handled by the getContext function

The specification states:

```
4.11.4 The canvas element
    ...
    The getContext(contextId, arguments...) method of the canvas element, when invoked,
    must run the steps in the cell of the following table whose column header describes
    the canvas element's canvas context mode and whose row header describes the method's
    first argument.
```

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

The null character `\0` is not properly handled by the `getContext` function.

V0219: The height and width content attribute values are not properly retrieved

The specification states:

```
4.11.4 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
    direct-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.
```

All document modes (All versions)

The `height` and `width` content attribute values are incorrectly truncating the content and not returning the value for `getAttribute`.

2.1.103 [HTML5] Section 4.11.4.2 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:

```
4.11.4.2 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.
```

All document modes (All versions)

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

2.1.104 [HTML5] Section 4.14.2 Pseudo-classes

V0221: The :active state is only set on one element, not on appropriate ancestor elements

The specification states:

```
4.14.2 Pseudo-classes
    ...
    :active

        The :active pseudo-class is defined to match an element “while an element is
        being activated by the user”. For the purposes of defining the :active
        pseudo-class only, an HTML user agent must consider an element as being activated
        if it is:
            ...
            An element that the user indicates using a pointing device while that
            pointing device is in the "down" state (e.g. for a mouse, between the time
            the mouse button is pressed and the time it is depressed).

            An element that has a descendant that is currently matching the :active
            pseudo-class.
```

All document modes (All versions)

The `:active` state is only set on one element and does not bubble up the element tree to properly activate all appropriate ancestor elements.

V0222: The :enabled state incorrectly identifies link elements that have no href attribute as enabled

The specification states:

```
4.14.2 Pseudo-classes
    ...
    :enabled

        The :enabled pseudo-class must match any element falling into one of the
        following categories:
            ...
            link elements that have an href attribute
```

All document modes (All versions)

The `:enabled` state incorrectly identifies `link` elements that have no `href` attribute as enabled.

V0223: The :enabled state does not identify a or area elements with href attributes as enabled

The specification states:

```
4.14.2 Pseudo-classes
    ...
    :enabled

        The :enabled pseudo-class must match any element falling into one of the
        following categories:
            ...
            a elements that have an href attribute

            area elements that have an href attribute

            link elements that have an href attribute
```

All document modes (All versions)

The `:enabled` state does not identify `a` or `area` elements with `href` attributes as enabled.

V0224: The :default state is not supported

The specification states:

```
4.14.2 Pseudo-classes
    ...
    :default

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

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

The `:default` state is not supported.

V0225: The :valid state does not properly handle form elements

The specification states:

```
4.14.2 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
```

All document modes (All versions)

The `:valid` state does not properly identify `form` elements as part of the `:valid` state when the form is a form owner of a valid candidate element.

V0226: The :valid state does not properly handle fieldset elements

The specification states:

```
4.14.2 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
```

All document modes (All versions)

The `valid` state does not properly identify `fieldset` elements as part of the `valid` state when the `fieldset` has a child that is a valid candidate element.

V0227: The :invalid state does not properly handle form elements

The specification states:

```
4.14.2 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
```

All document modes (All versions)

The `:invalid` state does not properly identify `form` elements as part of the `:invalid` state when the form is a form owner of an invalid candidate element.

V0228: The :invalid state does not properly handle fieldset elements

The specification states:

```
4.14.2 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
```

All document modes (All versions)

The `:invalid` state does not properly identify `fieldset` elements as part of the `:invalid` state when the `fieldset` has a child that is an invalid candidate element.

V0229: The :in-range and :out-of-range states are not supported

The specification states:

```
4.14.2 Pseudo-classes
    ...
    :in-range

        The :in-range pseudo-class must match all elements that are candidates for
        constraint validation, have range limitations, and that are neither suffering
        from an underflow nor suffering from an overflow.

    :out-of-range

        The :out-of-range pseudo-class must match all elements that are candidates for
        constraint validation, have range limitations, and that are either suffering from
        an underflow or suffering from an overflow.
```

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

The `:in-range` and `:out-of-range` states are not supported.

V0230: The :read-only and :read-write states are not supported

The specification states:

```
4.14.2 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.
```

All document modes (All versions)

The `:read-only` and `:read-write` states are not supported.

V0231: The :link selector does not identify area or link elements with href attributes

The specification states:

```
4.14.2 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.
```

All document modes (All versions)

The `:link` state does not identify `area` or `link` elements with `href` attributes.

V0232: The :indeterminate state does not match the input[type=radio] when no option is selected

The specification states:

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

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

All document modes (All versions)

The `:indeterminate` state does not match the `input[type=radio]` when no option is selected.

V0233: The :valid and :invalid states incorrectly match elements

The specification states:

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

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

            elements that are candidates for constraint validation and that satisfy their
            constraints

            form elements that are not the form owner of any elements that themselves are
            candidates for constraint validation but do not satisfy their constraints

            fieldset elements that have no descendant elements that themselves are
            candidates for constraint validation but do not satisfy their constraints

    :invalid

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

            elements that are candidates for constraint validation but that do not
            satisfy their constraints
```

```
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

fieldset elements that have of one or more descendant elements that
themselves are candidates for constraint validation but do not satisfy their
constraints
```

All document modes (All versions)

The `:valid` and `:invalid` states incorrectly match elements when the constraints are initially violated by the initial value defined on the element.

2.1.105 [HTML5] Section 5.1 Browsing contexts

V0234: The first Document is for the home page specified in the preferences, not for about:blank.

The specification states:

```
5.1 Browsing contexts
    ...
    When a browsing context is first created, it must be created with a single Document
    in its session history, whose address 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, along with a new Window object
    that the Document is associated with. The Document must have a single child html
    node, which itself has two empty child nodes: a head element, and a body element. As
    soon as this Document is created, the user agent must implement the sandboxing for
    it. If the browsing context has a creator Document, then the browsing context's
    Document's referrer must be set to the address of that creator Document at the time
    of the browsing context's creation.
```

All document modes (All versions)

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

V0235: A newly created browsing context does not change the readyState from loading to complete

The specification states:

```
5.1 Browsing contexts
    ...
    When a browsing context is first created, it must be created with a single Document
    in its session history, whose address 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, along with a new Window object
    that the Document is associated with. ...
```

All document modes (All versions)

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

V0236: Nested browsing contexts share a session history

The specification states:

```
5.5.1 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. ...
```

All document modes (All versions)

Nested browsing contexts share a session history.

V0237: In a newly created browsing context, the referrer is the empty string, not the creator document

The specification states:

```
5.1 Browsing contexts
    ...
    If the browsing context has a creator Document, then the browsing context's
    Document's referrer must be set to the address of that creator Document at the time
    of the browsing context's creation.
```

All document modes (All versions)

In a newly created browsing context, the referrer is the empty string, not the creator document.

V0238: The about:blank document does have a character encoding of UTF-8

The specification states:

```
5.1 Browsing contexts
    ...
    When a browsing context is first created, it must be created with a single Document
    in its session history, whose address 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, along with a new Window object
    that the Document is associated with. ...
```

All document modes (All versions)

The `about:blank` document does have a character encoding of UTF-8.

2.1.106 [HTML5] Section 5.1.1.1 Navigating nested browsing contexts in the DOM

V0239: In a nested browsing context, the frameElement attribute returns undefined, not an object

The specification states:

```
5.1.1.1 Navigating nested browsing contexts in the DOM
    ...
    The frameElement IDL attribute on the Window object of a Document d, on getting, must
```

```
run the following algorithm:

    1. If d is not a Document in a nested browsing context, return null and abort
       these steps.

    2. If the browsing context container's Document does not have the same effective
       script origin as the effective script origin specified by the entry settings
       object, then throw a SecurityError exception and abort these steps.

    3. Return the browsing context container for b.
```

All document modes (All versions)

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

V0240: The frameElement does not throw a security error

The specification states:

```
5.1.1.1 Navigating nested browsing contexts in the DOM
    ...
    The frameElement IDL attribute on the Window object of a Document d, on getting, must
    run the following algorithm:

        1. If d is not a Document in a nested browsing context, return null and abort
           these steps.

        2. If the browsing context container's Document does not have the same effective
           script origin as the effective script origin specified by the entry settings
           object, then throw a SecurityError exception and abort these steps.

        3. Return the browsing context container for b.
```

All document modes (All versions)

The `frameElement` does not throw a security error.

2.1.107 [HTML5] Section 5.1.6 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:

```
5.1.6 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.
```

All document modes (All versions)

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:

```
5.1.6 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.
```

All document modes (All versions)

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

2.1.108 [HTML5] Section 5.2 The Window object

V0243: The stop function is not supported

The specification states:

```
5.2 The Window object

    [Global]
    /*sealed*/ interface Window : EventTarget {
        ...
        void stop();
        ...
    };
```

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

The `stop` function is not supported.

V0244: The locationbar, menubar, personalbar, scrollbars, statusbar, and toolbar attributes are not supported

The specification states:

```
5.2 The Window object

    [Global]
    /*sealed*/ interface Window : EventTarget {
        ...
        [Replaceable] readonly attribute BarProp locationbar;
        [Replaceable] readonly attribute BarProp menubar;
        [Replaceable] readonly attribute BarProp personalbar;
        [Replaceable] readonly attribute BarProp scrollbars;
        [Replaceable] readonly attribute BarProp statusbar;
        [Replaceable] readonly attribute BarProp toolbar;
        ...
    };
```

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

The `locationbar`, `menubar`, `personalbar`, `scrollbars`, `statusbar` and `toolbar` attributes are not supported.

V0245: The HTMLDocument property is not supported

The specification states:

```
5.2 The Window object
    ...
    For historical reasons, Window objects must also have a writable, configurable,
    non-enumerable property named HTMLDocument whose value is the Document interface
    object.
```

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

The `HTMLDocument` property is not supported.

2.1.109 [HTML5] Section 5.2.1 Security

V0246: Many attributes do not throw a SecurityError exception when accessed from a different origin

The specification states:

```
5.2.1 Security
    ...
    User agents must throw a SecurityError exception whenever any properties of a Window
    object are accessed when the incumbent settings object specifies an effective script
    origin that is not the same as the Window object's Document's effective script
    origin, with the following exceptions:
```

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

Attributes `applicationCache`, `document`, `external`, `frameElement`, `history`, `locationbar`, `menubar`, `name`, `navigator`, `personalbar`, `scrollbars`, `statusbar`, `status`, and `toolbar` do not throw a `SecurityError` exception when accessed from a different origin.

V0247: Many functions do not throw SecurityError exception when accessed from a different origin

The specification states:

```
5.2.1 Security
    ...
    User agents must throw a SecurityError exception whenever any properties of a Window
    object are accessed when the incumbent settings object specifies an effective script
    origin that is not the same as the Window object's Document's effective script
    origin, with the following exceptions:
```

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

The functions `alert`, `confirm`, `open`, `print`, `prompt` and `stop` do not throw a `SecurityError` exception when accessed from a different origin.

V0248: Events do not throw a SecurityError exception when accessed from a different origin

The specification states:

```
5.2.1 Security
    ...
    User agents must throw a SecurityError exception whenever any properties of a Window
    object are accessed when the incumbent settings object specifies an effective script
    origin that is not the same as the Window object's Document's effective script
    origin, with the following exceptions:
```

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

Events do not throw a `SecurityError` exception when accessed from a different origin

2.1.110 [HTML5] Section 5.2.2 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:

```
5.2.2 APIs for creating and navigating browsing contexts by name
    ...
    If applying the rules for choosing a browsing context given a browsing context name
    using the target argument would result in there not being a chosen browsing context,
    then throw an InvalidAccessError exception and abort these steps.
```

All document modes (All versions)

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

V0250: The stop function is not supported

The specification states:

```
5.2.2 APIs for creating and navigating browsing contexts by name
    ...
    The stop() method on Window objects should, if there is an existing attempt to
    navigate the browsing context and that attempt is not currently running the unload a
    document algorithm, cancel that navigation; then, it must abort the active document
    of the browsing context of the Window object on which it was invoked.
```

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

The `stop` function is not supported.

2.1.111 [HTML5] Section 5.2.3 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 on the Window interface must return the number of child
    browsing contexts that are nested through elements that are in the Document that is
    the active document of that Window object, if that Window's browsing context shares
    the same event loop as the responsible document specified by the entry settings
    object accessing the IDL attribute; otherwise, it must return zero.
```

All document modes (All versions)

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

V0252: Accessing window objects by index ignores descendant browsing contexts

The specification states:

```
5.2.3 Accessing other browsing contexts
    ...
    To determine the value of an indexed property index of a Window object, the user
    agent must return the WindowProxy object of the index'th child browsing context of
    the Document that is nested through an element that is in the Document, sorted in the
    tree order of the elements nesting those browsing contexts.
```

All document modes (All versions)

Accessing window objects by index ignores descendant browsing contexts.

2.1.112 [HTML5] Section 5.2.4 Named access on the Window object

V0253: Framesets are not identifiable by name

The specification states:

```
5.2.4 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
        ...
```

All document modes (All versions)

Framesets are not identifiable by name.

2.1.113 [HTML5] Section 5.3.1 Relaxing the same-origin restriction

V0254: No SecurityError exceptiont is thrown if there is no browsing context et al.

The specification states:

```
5.3.1 Relaxing the same-origin restriction
    ...
    On setting, the user agent must run the following algorithm:

        1.  If the Document has no browsing context, throw a SecurityError exception and
            abort these steps.

        2.  If the Document's active sandboxing flag set has its sandboxed
            document.domain browsing context flag set, throw a SecurityError exception
            and abort these steps.

        3.  If the new value is an IPv4 or IPv6 address, let new value be the new value.
            Otherwise, apply the IDNA ToASCII algorithm to the new value, with both the
            AllowUnassigned and UseSTD3ASCIIRules flags set, and let new value be the
            result of the ToASCII algorithm.

            If ToASCII fails to convert one of the components of the string, e.g. because
            it is too long or because it contains invalid characters, then throw a
            SecurityError exception and abort these steps. [RFC5890]

        4. If new value is not exactly equal to the current value of the document.domain
           attribute, then run these substeps:

            1.  If the current value is an IPv4 or IPv6 address, throw a SecurityError
                exception and abort these steps.

            2.  If new value, prefixed by a "." (U+002E), does not exactly match the end
                of the current value, throw a SecurityError exception and abort these
                steps.

                Note: If the new value is an IPv4 or IPv6 address, it cannot match the
                new value in this way and thus an exception will be thrown here.

            3.  If new value matches a suffix in the Public Suffix List, or, if new
                value, prefixed by a "." (U+002E), matches the end of a suffix in the
                Public Suffix List, then throw a SecurityError exception and abort these
```

```
steps. [PSL]
```

All document modes (All versions)

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`.

2.1.114 [HTML5] Section 5.5.3.1 Security

V0255: The 'Location' object does not properly throw a Security Exception

The specification states:

```
5.5.3.1 Security
    ...
    User agents must throw a SecurityError exception whenever any properties of a
    Location object are accessed when the entry settings object specifies an effective
    script origin that is not the same as the Location object's associated Document's
    browsing context's active document's effective script origin, with the following
    exceptions:
        ...
```

All document modes (All versions)

The `Location` object does not properly throw a Security Exception and no exception is thrown.

2.1.115 [HTML5] Section 5.6.6 Page load processing model for media

V0256: Audio and video media do not load into a Document but attempt to download instead

The specification states:

```
5.6.6 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 sniffed 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.
```

All document modes (All versions)

Audio and video media do not load into a `Document` but attempt to download instead.

2.1.116 [HTML5] Section 5.6.7 Page load processing model for content that uses plugins

V0257: Plugins do not load into a newly created document and attempt to download instead

The specification states:

```
5.6.7 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
    sniffed 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.
```

All document modes (All versions)

Plugins do not load into a newly created document and attempt to download instead.

2.1.117 [HTML5] Section 5.6.10.1 Event definitions

V0258: The PopStateEventInit dictionary is not supported

The specification states:

```
5.6.10.1 Event definitions
    ...
    dictionary PopStateEventInit : EventInit {
        ...
    };
```

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

The `PopStateEventInit` dictionary is not supported.

V0259: The HashChangeEvent interface is not supported

The specification states:

```
5.6.10.1 Event definitions
    ...
    interface HashChangeEvent : Event {
        ...
    };
```

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

The `HashChangeEvent` interface is not supported.

V0260: The HashChangeEventInit dictionary is not supported

The specification states:

```
5.6.10.1 Event definitions
    ...
    dictionary HashChangeEventInit : EventInit {
        ...
    };
```

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

The `HashChangeEventInit` dictionary is not supported.

V0261: The PageTransitionEventInit dictionary is not supported

The specification states:

```
5.6.10.1 Event definitions
    ...
    dictionary PageTransitionEventInit : EventInit {
        ...
    };
```

All document modes (All versions)

The `PageTransitionEventInit` dictionary is not supported.

2.1.118 [HTML5] Section 5.6.11 Unloading documents

V0262: The open function does not increment the ignore-opens-during-unload counter during unload of a document

The specification states:

```
5.6.11 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.
```

All document modes (All versions)

The `open` function does not increment the `ignore-opens-during-unload` counter during unload of a document.

2.1.119 [HTML5] Section 5.7.3.3 Parsing cache manifests

V0263: The settings parse mode is not supported

The specification states:

```
5.7.3.3 Parsing cache manifests
    ...
```

```
When a user agent is to parse a manifest, it means that the user agent must run the
following steps:
    ...
    30.  Process tokens as follows:
            ...
            If mode is "settings"

                If tokens contains a single token, and that token is a case-sensitive
                match for the string "prefer-online", then set cache mode flag to
                prefer-online and jump back to the step labeled start of line.

                Otherwise, the line is an unsupported setting: do nothing; the line
                is ignored.
```

All document modes (All versions)

The `settings` parse mode is not supported.

2.1.120 [HTML5] Section 6.1.3.6 Runtime script errors

V0264: Muted errors functionality is not supported

The specification states:

```
6.1.3.6 Runtime script errors
    ...
    When the user agent is required to report an error for a particular script script
    with a particular position line:col, using a particular target target, it must run
    these steps, after which the error is either handled or not handled:
        ...
        6.  If script has muted errors, then set message to "Script error.", set location
            to the empty string, set line and col to 0, and set error object to null.
```

All document modes (All versions)

Muted errors functionality is not supported because cross-origin restrictions are not enforced for scripts.

2.1.121 [HTML5] Section 6.1.3.6.2 The ErrorEvent interface

V0265: The ErrorEvent interface is not supported

The specification states:

```
6.1.3.6.2 The ErrorEvent interface
    ...
    interface ErrorEvent : Event {
        ...
    };
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (Internet Explorer 9 and Internet Explorer 8)

The `errorEvent` interface is not supported.

V0266: The colno attribute of the ErrorEventInit interface is not supported

The specification states:

```
6.1.3.6.2 The ErrorEvent interface
    ...
    dictionary ErrorEventInit : EventInit {
        ...
        unsigned long colno;
        ...
    };
```

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

The `colno` attribute of the `ErrorEventInit` interface is not supported.

V0267: The ErrorEventInit dictionary is not supported

The specification states:

```
6.1.3.6.2 The ErrorEvent interface
    ...
    dictionary ErrorEventInit : EventInit {
        ...
    };
```

IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `ErrorEventInit` dictionary is not supported

2.1.122 [HTML5] Section 6.1.5.1 Event handlers

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

The specification states:

```
6.1.5.1 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.
```

All document modes (All versions)

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

V0269: Removing an event content attribute does not remove the content attribute specified value from the event

The specification states:

```
6.1.5.1 Event handlers
    ...
    When an event handler content attribute is removed, the user agent must set the
    corresponding event handler to null.
```

IE11 mode (All versions)

Removing an event content attribute does not remove the content attribute specified value from the event.

V0270: The error argument is not supported on the OnErrorEventHandlerNonNull callback function.

The specification states:

```
6.1.5.1 Event handlers
    ...
    [TreatNonCallableAsNull]
    callback OnErrorEventHandlerNonNull = any ((Event or DOMString) event, optional
    DOMString source, optional unsigned long lineno, optional unsigned long column,
    optional any error);
```

All document modes (All versions)

The `error` argument is not supported on the `OnErrorEventHandlerNonNull` callback function.

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

The specification states:

```
6.1.5.1 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.
```

All document modes (All versions)

The form owner is not taken into account in events.

2.1.123 [HTML5] Section 6.1.5.2 Event handlers on elements, Document objects, and Window objects

V0272: The oncancel event handler is not supported

The specification states:

```
6.1.5.2 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
```

All document modes (All versions)

The `oncancel` event handler is not supported.

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

The specification states:

```
6.1.5.2 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
```

All document modes (All versions)

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

V0274: The oninvalid event handler is not supported

The specification states:

```
6.1.5.2 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:
        ...
        oninvalid    invalid
```

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

The `oninvalid` event handler is not supported.

V0275: The onmouseenter and onmouseleave event handlers are not supported on Document or Window

The specification states:

```
6.1.5.2 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
```

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

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

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

The specification states:

```
6.1.5.2 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
```

All document modes (All versions)

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

V0277: The onshow event handler is not supported

The specification states:

```
6.1.5.2 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
```

All document modes (All versions)

The `onshow` event handler is not supported.

V0278: The ontoggle event handler is not supported

The specification states:

```
6.1.5.2 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
```

All document modes (All versions)

The `ontoggle` event handler is not supported.

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

The specification states:

```
6.1.5.2 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
```

All document modes (All versions)

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:

```
6.1.5.2 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 (All versions)

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

2.1.124 [HTML5] Section 6.3.1 Opening the input stream

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

The specification states:

```
6.3.1 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.
```

All document modes (All versions)

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

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

The specification states:

```
6.3.1 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.
```

All document modes (All versions)

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

V0282: Singleton objects are not replaced

The specification states:

```
6.3.1 Opening the input stream
    ...
    When called with two arguments, the document.open() method must act as follows:
        ...
        15.  Replace the Document's singleton objects with new instances of those
             objects. (This includes in particular the Window, Location, History,
             ApplicationCache, and Navigator, objects, the various BarProp objects, the
             two Storage objects, the various HTMLCollection objects, and objects defined
             by other specifications, like Selection and the document's UndoManager. It
             also includes all the Web IDL prototypes in the JavaScript binding,
             including the Document object's prototype.)
```

All document modes (All versions)

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

V0283: The script-created parser is incorrectly freed from the script stack

The specification states:

```
.3.1 Opening the input stream
    ...
    When called with two arguments, the document.open() method must act as follows:
        ...
        21. 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.
```

All document modes (All versions)

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

2.1.125 [HTML5] Section 6.3.2 Closing the input stream

V0284: Non-HTML documents do not throw an InvalidStateError exception

The specification states:

```
6.3.2 Closing the input stream
    ...
    The close() method must run the following steps:
        ...
        1.  If the Document object is not flagged as an HTML document, throw an
            InvalidStateError exception and abort these steps.
```

All document modes (All versions)

Non-HTML documents do not throw an `InvalidStateError` exception.

2.1.126 [HTML5] Section 6.3.3 document.write()

V0285: Non-HTML documents do not throw InvalidStateError exception

The specification states:

```
6.3.3 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.
```

All document modes (All versions)

Non-HTML documents do not throw an `InvalidStateError` exception.

2.1.127 [HTML5] Section 6.6.1 The Navigator object

V0286: The NavigatorLanguage interface is not supported

The specification states:

```
6.6.1 The Navigator object
    ...
    interface Navigator {
        // objects implementing this interface also implement the interfaces given below
    };
    ...
    Navigator implements NavigatorLanguage;
```

All document modes (All versions)

The `NavigatorLanguage` interface is not supported.

V0287: The NavigatorPlugins interface is not supported

The specification states:

```
6.6.1 The Navigator object
    ...
    interface Navigator {
        // objects implementing this interface also implement the interfaces given below
    };
    ...
    Navigator implements NavigatorPlugins;
```

All document modes (All versions)

The `NavigatorPlugins` interface is not supported.

2.1.128 [HTML5] Section 6.6.1.1 Client identification

V0288: The taintEnabled function is not supported

The specification states:

```
6.6.1 The Navigator object
    ...
    [NoInterfaceObject]
    interface NavigatorID {
        ...
        boolean taintEnabled(); // constant false
        ...
     };
```

All document modes (All versions)

The `taintEnabled` function is not supported.

V0289: The appCodeName is implemented directly on the Navigator interface, not on the NavigatorID interface

The specification states:

```
6.6.1 The Navigator object
    ...
    [NoInterfaceObject]
    interface NavigatorID {
       readonly attribute DOMString appCodeName; // constant "Mozilla"
        ...
     };
```

All document modes (All versions)

The `appCodeName` attribute is implemented directly on the `Navigator` interface, not on the `NavigatorID` interface.

V0290: The product attribute is not supported

The specification states:

```
6.6.1 The Navigator object
    ...
    [NoInterfaceObject]
    interface NavigatorID {
        ...
       readonly attribute DOMString product; // constant "Gecko"
        ...
     };
```

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

The `product` attribute is not supported.

2.1.129 [HTML5] Section 6.6.1.2 Language preferences

V0291: The language attribute is implemented directly on the Navigator interface, not on the NavigatorLanguage interface

The specification states:

```
6.6.1.2 Language preferences

    [NoInterfaceObject]
    interface NavigatorLanguage {
        readonly attribute DOMString? language;
    };
```

All document modes (All versions)

The `language` attribute is implemented directly on the `Navigator` interface, not on the `NavigatorLanguage` interface.

2.1.130 [HTML5] Section 6.6.1.3 Custom scheme and content handlers

V0292: The NavigatorContentUtils interface is implemented but does not support any functions

The specification states:

```
6.6.1.3 Custom scheme and content handlers

    [NoInterfaceObject]
    interface NavigatorContentUtils {
        // content handler registration
        void registerProtocolHandler(DOMString scheme, DOMString url, DOMString title);
        void registerContentHandler(DOMString mimeType, DOMString url, DOMString title);
        void unregisterProtocolHandler(DOMString scheme, DOMString url);
        void unregisterContentHandler(DOMString mimeType, DOMString url);
    };
```

All document modes (All versions)

The `NavigatorContentUtils` interface is implemented but does not support any functions.

2.1.131 [HTML5] Section 6.6.1.4 Manually releasing the storage mutex

V0293: The cookieEnabled attribute is defined on the Navigator interface directly, not on NavigatorStorageUtils

The specification states:

```
6.6.1.4 Manually releasing the storage mutex

    [NoInterfaceObject]
    interface NavigatorStorageUtils {
        readonly attribute boolean cookieEnabled;
        ...
    };
```

All document modes (All versions)

The `cookieEnabled` attribute is defined on the `Navigator` interface directly, not on `NavigatorStorageUtils`.

V0294: The yieldForStorageUpdated function is not supported

The specification states:

```
6.6.1.4 Manually releasing the storage mutex

    [NoInterfaceObject]
    interface NavigatorStorageUtils {
        ...
        void yieldForStorageUpdates();
    };
```

All document modes (All versions)

The `yieldForStorageUpdated` function is not supported.

2.1.132 [HTML5] Section 6.6.1.5 Plugins

V0295: The NavigatorPlugins interface is not supported

The specification states:

```
6.6.1.5 Plugins

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

All document modes (All versions)

The `NavigatorPlugins` interface is not supported.

V0296: The PluginArray, Plugin, MimeTypeArray and MimeType interfaces are not supported

The specification states:

```
6.6.1.5 Plugins

    interface PluginArray {
        ...
    };

    interface MimeTypeArray {
        ...
    };

    interface Plugin {
        ...
    };

    interface MimeType {
        ...
    };
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

The `PluginArray`, `MimeTypeArray`, `Plugin`, and `MimeType` interfaces are not supported.

2.1.133 [HTML5] Section 7.4.1 Sequential focus navigation and the tabindex attribute

V0297: tabindex incorrectly returns [InlineCode|0] as the default for elements that are not focusable

The specification states:

```
7.4.1 Sequential focus navigation and the tabindex attribute
```

```
...
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.
```

All document modes (All versions)

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

2.1.134 [HTML5] Section 7.6.1 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:

```
7.6.1 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).
```

All document modes (All versions)

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

V0299: The contentEditable attribute does not throw a DOMException SYNTAX_ERR

The specification states:

```
7.6.1 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 exception.
```

All document modes (All versions)

An invalid value for the `contentEditable` attribute does not throw a `DOMException SYNTAX_ERR`, but instead throws an "Invalid Argument" exception.

2.1.135 [HTML5] Section 7.6.2 Making entire documents editable: The designMode IDL attribute

V0300: The designMode attribute incorrectly has a third state of inherit that should be considered invalid

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.
```

All document modes (All versions)

The `designMode` attribute incorrectly has a third state of `inherit` that should be considered invalid.

2.1.136 [HTML5] Section 7.6.5 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:

```
7.6.5 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.
```

All document modes (All versions)

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

2.1.137 [HTML5] 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.
```

All document modes (All versions)

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

2.1.138 [HTML5] Section 8.2.3.1 The insertion mode

V0303: The "in template" insertion mode is not implemented because the template element is not supported

The specification states:

```
8.2.3.1 The insertion mode
    ...
    Initially, the insertion mode is "initial". It can change to "before html", "before
    head", "in head", "in head noscript", "after head", "in body", "text", "in table",
    "in table text", "in caption", "in column group", "in table body", "in row", "in
    cell", "in select", "in select in table", "in template", "after body", "in frameset",
    "after frameset", "after after body", and "after after frameset" during the course of
    the parsing, as described in the tree construction stage. The insertion mode affects
    how tokens are processed and whether CDATA sections are supported.
```

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

The "in template" insertion mode is not implemented because the `template` element is not supported.

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
    ...
    When the steps below require the UA to reset the insertion mode appropriately, it
    means the UA must follow these steps:
        ...
        12.  If node is a head element and last is true, then switch the insertion mode
             to "in body" ("in body"! not "in head"!) and abort these steps. (fragment
             case)

        13.  If node is a head element and last is false, then switch the insertion mode
             to "in head" and abort these steps.
```

All document modes (All versions)

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

2.1.139 [HTML5] 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
```

All document modes (All versions)

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

2.1.140 [HTML5] Section 8.2.4.38 Attribute value (double-quoted) state

V0306: NULL character U+0000 does not produce a parse error

The specification states:

```
8.2.4.38 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.
```

All document modes (All versions)

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

2.1.141 [HTML5] Section 8.2.4.39 Attribute value (single-quoted) state

V0307: NULL character U+0000 does not produce a parse error

The specification states:

```
8.2.4.39 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.
```

All document modes (All versions)

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

2.1.142 [HTML5] Section 8.2.4.45 Markup declaration open state

V0308: A non-HTML namespace CDATA section is not consumed properly

The specification states:

```
8.2.4.45 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.
```

All document modes (All versions)

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

2.1.143 [HTML5] Section 8.2.4.48 Comment state

V0309: There is no parse error when a U+0000 NULL character is encountered

The specification states:

```
8.2.4.48 Comment state

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

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

All document modes (All versions)

There is no parse error when a U+0000 NULL character is encountered.

2.1.144 [HTML5] 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:

        ▪ An mi element in the MathML namespace
        ▪ An mo element in the MathML namespace
        ▪ An mn element in the MathML namespace
        ▪ An ms element in the MathML namespace
        ▪ An mtext element in the MathML namespace
```

All document modes (All versions)

MathML is not supported.

2.1.145 [HTML5] 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 option element, an
    optgroup 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.
```

All document modes (All versions)

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

2.1.146 [HTML5] Section 8.2.5.4.7 The "in body" insertion mode

V0312: The rule of three active formatting elements is not implemented

The specification states:

```
8.2.5.4.7 The "in body" insertion mode
    ...
    The adoption agency algorithm, which takes as its only argument a tag name subject
    for which the algorithm is being run, consists of the following steps:
        ...
        13.  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.
```

All document modes (All versions)

The rule of three active formatting elements is not implemented (substep 5 of step 13).

2.1.147 [HTML5] 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.
```

All document modes (All versions)

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

2.1.148 [HTML5] Section 8.2.5.4.11 The "in caption" insertion mode

V0314: Tags in an open tag do not properly pop elements off the stack

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"

            Parse error.

            If the stack of open elements does not have a caption element in table scope,
            ignore the token. (fragment case)

            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.
```

All document modes (All versions)

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.

2.1.149 [HTML5] 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

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.
```

All document modes (All versions)

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.

2.1.150 [HTML5] Section 8.2.5.4.18 The "in template" insertion mode

V0316: The template element is not supported

The specification states:

```
8.2.5.4.18 The "in template" insertion mode

    When the user agent is to apply the rules for the "in template" insertion mode, the
    user agent must handle the token as follows:
```

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

The `template` element is not supported.

2.1.151 [HTML5] 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.
```

All document modes (All versions)

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

2.1.152 [HTML5] 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;
    }
```

All document modes (All versions)

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

V0319: The noframes element's default style 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;
    }
```

All document modes (All versions)

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

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

The specification states:

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

All document modes (All versions)

The `embed` element when hidden does not set the default styles and only hides the element.

2.1.153 [HTML5] 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;
    }
```

All document modes (All versions)

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

V0322: The listing, plaintext, and xmp elements do not set top or bottom margins in the default styles

The specification states:

```
10.3.3 Flow content
    ...
    blockquote, figure, listing, p, plaintext, pre, xmp {
        margin-top: 1em; margin-bottom: 1em;
    }
```

All document modes (All versions)

The `listing`, `plaintext`, and `xmp` elements do not set top or bottom margins in the default styles.

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;
```

```
}
```

All document modes (All versions)

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; }
```

All document modes (All versions)

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

2.1.154 [HTML5] Section 10.3.4 Phrasing content

V0325: The b and strong elements do not set the correct font-weight: bolder value in the default styles

The specification states:

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

All document modes (All versions)

The `b` and `strong` elements do not set the correct `font-weight: bolder` value in the default styles but instead sets `font-weight: bold`.

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; }
```

All document modes (All versions)

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

V0327: The elements small, sub, and sup do not properly 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; }
```

All document modes (All versions)

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; }
```

All document modes (All versions)

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

V0329: The element rt does not set some properties in the default styles

The specification states:

```
10.3.4 Phrasing content
    ...
    rt   {
        display: ruby-text;
        white-space: nowrap;
        font-size: 50%;
        font-variant-east-asian: ruby;
        text-emphasis: none;
    }
```

All document modes (All versions)

The element `rt` does not set `white-space`, `font-variant-east-asian`, and `text-emphasis` in the default styles.

V0330: The elements ruby, rb, rt, rbc, and rtc do not set unicode-bidi: isolate in the default styles

The specification states:

```
10.3.4 Phrasing content
    ...
    ruby, rb, rt, rbc, rtc { unicode-bidi: isolate; }
```

All document modes (All versions)

The elements `ruby`, `rb`, `rt`, `rbc`, and `rtc` do not set `unicode-bidi: isolate` in the default styles.

V0331: The :link and :visited states do not set 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; }
```

All document modes (All versions)

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; }
```

All document modes (All versions)

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; }
```

All document modes (All versions)

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

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

The specification states:

```
10.3.4 Phrasing content
    ...
    br { content: '\A'; white-space: pre; }
    nobr { white-space: nowrap; }
    wbr { content: '\200B'; }
    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; }
```

All document modes (All versions)

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

V0335: The font size attribute incorrectly maps values

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'.
```

All document modes (All versions)

The font size attribute incorrectly maps the values as follows:

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

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

1     | xx-small            |

2     | x-small             |

3     | small               |
```

```
4     | medium              |
5     | large               |
6     | x-large             |
7     | xx-large            |
```

2.1.155 [HTML5] Section 10.3.5 Bidirectional text

V0336: 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]:dir(ltr) { direction: ltr; }
    [dir]:dir(rtl), bdi:dir(rtl) { direction: rtl; }

    address, blockquote, center, div, figure, figcaption, footer, form,
    header, hr, legend, listing, p, plaintext, pre, xmp, article,
    aside, h1, h2, h3, h4, h5, h6, hgroup, main, 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; }

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

All document modes (All versions)

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

2.1.156 [HTML5] Section 10.3.6 Quotes

V0337: All quote values are not defined in the default styles

The specification states:

```
10.3.6 Quotes
```

All document modes (All versions)

All quote values are not defined in the default styles.

2.1.157 [HTML5] Section 10.3.7 Sections and headings

V0338: Nesting rules for sections and headings are not defined

The specification states:

```
10.3.7 Sections and headings
    ...
    The article, aside, nav, and section elements are expected to affect the margins and
    font size of h1 elements, as well as h2–h5 elements that follow h1 elements in hgroup
    elements, based on the nesting depth. If x is a selector that matches elements that
    are either article, aside, nav, or section elements, then the following rules capture
    what is expected:

    @namespace url(http://www.w3.org/1999/xhtml);

    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; }

    x hgroup > h1 ~ h2 { margin-top: 1.00em; margin-bottom: 1.00em; font-size: 1.17em; }
    x x hgroup > h1 ~ h2 { margin-top: 1.33em; margin-bottom: 1.33em; font-size: 1.00em; }
    x x x hgroup > h1 ~ h2 { margin-top: 1.67em; margin-bottom: 1.67em; font-size:
    0.83em; }
    x x x x hgroup > h1 ~ h2 { margin-top: 2.33em; margin-bottom: 2.33em; font-size:
    0.67em; }

    x hgroup > h1 ~ h3 { margin-top: 1.33em; margin-bottom: 1.33em; font-size: 1.00em; }
    x x hgroup > h1 ~ h3 { margin-top: 1.67em; margin-bottom: 1.67em; font-size: 0.83em; }
    x x x hgroup > h1 ~ h3 { margin-top: 2.33em; margin-bottom: 2.33em; font-size:
    0.67em; }

    x hgroup > h1 ~ h4 { margin-top: 1.67em; margin-bottom: 1.67em; font-size: 0.83em; }
    x x hgroup > h1 ~ h4 { margin-top: 2.33em; margin-bottom: 2.33em; font-size: 0.67em; }

    x hgroup > h1 ~ h5 { margin-top: 2.33em; margin-bottom: 2.33em; font-size: 0.67em; }
```

All document modes (All versions)

Nesting rules for sections and headings are not defined.

2.1.158 [HTML5] Section 10.3.8 Lists

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

The specification states:

```
10.3.8 Lists
    ...
    dd { margin-left: 40px; } /* LTR-specific: use 'margin-right' for rtl elements */
```

All document modes (All versions)

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

V0340: The dl element does not set margins within the default styles

The specification states:

```
10.3.8 Lists
```

```
    ...
    dir, dl, ol, ul { margin-top: 1em; margin-bottom: 1em; }

    dir dir, dir dl, dir ol, dir ul,
    dl dir, dl dl, dl ol, dl ul,
    ol dir, ol dl, ol ol, ol ul,
    ul dir, ul dl, ul ol, ul ul {
        margin-top: 0; margin-bottom: 0;
    }
```

All document modes (All versions)

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

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; }
```

All document modes (All versions)

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

2.1.159 [HTML5] 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;
    }
```

All document modes (All versions)

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;
    }
```

All document modes (All versions)

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.

V0344: The th element does not set font-weight: bold to the correct value in default styles

The specification states:

```
10.3.9 Tables
    ...
    th { font-weight: bold; }
```

All document modes (All versions)

The `th` element does not set `font-weight: bold` value in default styles.

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;
    border-style: none 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-left-width: 1px;
    border-left-style: solid;
    border-right-width: 1px;
    border-right-style: solid;
}
table[rules=groups i] > thead,
table[rules=groups i] > tbody,
table[rules=groups i] > tfoot {
    border-top-width: 1px;
    border-top-style: solid;
    border-bottom-width: 1px;
    border-bottom-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-top-width: 1px;
    border-top-style: solid;
    border-bottom-width: 1px;
    border-bottom-style: solid;
```

```
    }
```

All document modes (All versions)

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

V0346: The default styles for the table element are not defined

The specification states:

```
10.3.9 Tables
    ...
    table {
        font-weight: initial;
        font-style: initial;
        font-variant: initial;
        font-size: initial;
        line-height: initial;
        white-space: initial;
        text-align: initial;
    }
```

IE5 (Quirks) mode (All versions)

The default styles for the `table` element are not defined.

V0347: A form element within a table, thead, tbody, tfoot, or tr element is not set to display: none

The specification states:

```
10.3.9 Tables
    ...
    In HTML documents, the user agent is expected to force the 'display' property of form
    elements that are children of table, thead, tbody, tfoot, or tr elements to compute
    to 'none', irrespective of CSS rules.
```

All document modes (All versions)

A `form` element within a `table`, `thead`, `tbody`, `tfoot`, or `tr` element is not set to `display: none`.

V0348: Background images on table elements are not aligned 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 resolved relative to the
    element, 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 absolute URL.
```

All document modes (All versions)

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

2.1.160 [HTML5] 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;
    }
```

All document modes (All versions)

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

V0350: All input controls are set to box-sizing: border-box

The specification states:

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

All document modes (All versions)

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

V0351: The input elements other than type=image and textarea do not set box-sizing: border-box in the default styles

The specification states:

```
10.3.11 Form controls
    ...
    In quirks mode, the following rules are also expected to apply:
        @namespace url(http://www.w3.org/1999/xhtml);

        input:not([type=image]), textarea { box-sizing: border-box; }
```

IE5 (Quirks) mode (All versions)

The `input` elements other than `type=image` and `textarea` elements do not set `box-sizing: border-box` in the default styles.

2.1.161 [HTML5] Section 10.3.12 The hr element

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: 0.5em auto; }
```

All document modes (All versions)

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`.

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

The specification states:

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

All document modes (All versions)

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

2.1.162 [HTML5] 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 {
        margin-left: 2px; margin-right: 2px;
        border: groove 2px ThreeDFace;
        padding: 0.35em 0.625em 0.75em;
    }
```

All document modes (All versions)

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

2.1.163 [HTML5] 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; }
```

All document modes (All versions)

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

V0356: The default border style for the iframe element is border-style: none in default styles

The specification states:

```
10.4.1 Embedded content
    ...
    The following CSS rules are expected to apply:

        iframe { border: 2px inset; }
```

All document modes (All versions)

The default border style for the `iframe` element is `border-style: none` in default styles.

2.1.164 [HTML5] 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.
```

All document modes (All versions)

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

2.1.165 [HTML5] 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;
    }
```

All document modes (All versions)

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

2.1.166 [HTML5] 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.
```

All document modes (All versions)

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

2.1.167 [HTML5] Section 10.5 Bindings

V0360: Bindings are not supported

The specification states:

```
10.5 Bindings

    10.5.1 Introduction

        A number of elements have their rendering defined in terms of the 'binding'
        property.
```

All document modes (All versions)

Bindings are not supported; similar functionality is provided through non-CSS means.

2.1.168 [HTML5] Section 10.5.11 The meter element

V0361: The meter element is not supported

The specification states:

```
4.10.15 The meter element
    ...
    The meter element represents a scalar measurement within a known range, or a
    fractional value; for example disk usage, the relevance of a query result, or the
    fraction of a voting population to have selected a particular candidate.
```

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

The `meter` element is not supported.

2.1.169 [HTML5] Section 11.3.4 Other elements, attributes and APIs

V0363: The listing, plaintext, and xmp elements are not implemented on the HTMLElement interface

The specification states:

```
11.3.4 Other elements, attributes and APIs
    ...
    User agents must treat listing elements in a manner equivalent to pre elements in
    terms of semantics and for purposes of rendering.
    ...
    User agents must treat plaintext elements in a manner equivalent to pre elements in
    terms of semantics and for purposes of rendering. (The parser has special behavior
    for this element, though.)
    ...
    User agents must treat xmp elements in a manner equivalent to pre elements in terms
    of semantics and for purposes of rendering. (The parser has special behavior for this
    element though.)
```

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

The `listing`, `plaintext` and `xmp` elements are implemented on the `HTMLBlockElement` interface, not on `HTMLElement`.

V0364: The body element attributes text, link, vLink, aLink, and bgColor return color values as hex color values

The specification states:

```
11.3.4 Other elements, attributes and APIs
    ...
    The text IDL attribute of the body element must reflect the element's text content
    attribute.

    The link IDL attribute of the body element must reflect the element's link content
    attribute.

    The aLink IDL attribute of the body element must reflect the element's alink content
    attribute.

    The vLink IDL attribute of the body element must reflect the element's vlink content
    attribute.

    The bgColor IDL attribute of the body element must reflect the element's bgcolor
    content attribute.
```

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

The `body` element attributes `text`, `link`, `vLink`, `aLink`, and `bgColor` return color values as hex color values, not as specified.

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

The specification states:

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

All document modes (All versions)

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

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

The specification states:

```
11.3.4 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.
```

All document modes (All versions)

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

V0367: The lowsrc attribute is not implemented on the HTMLImageElement interface

The specification states:

```
11.3.4 Other elements, attributes and APIs
    ...
    The lowsrc IDL attribute of the img element must reflect the element's lowsrc content
    attribute, which for the purposes of reflection is defined as containing a URL.
```

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

The `lowsrc` attribute is not implemented on the HTMLImageElement interface.

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

The specification states:

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

All document modes (All versions)

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

V0369: The pre element does not convert the width attribute to a type long

The specification states:

```
11.3.4 Other elements, attributes and APIs

    partial interface HTMLPreElement {
        attribute long width;
    };

    The width IDL attribute of the pre element must reflect the content attribute of the
    same name.
```

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

The `pre` element does not convert the `width` attribute to a type long.

V0370: The blink element is of type HTMLPhraseElement, not HTMLUnknownElement

The specification states:

```
11.3.4 Other elements, attributes and APIs
    ...
    The blink, bgsound, isindex, multicol, nextid, and spacer elements must use the
    HTMLUnknownElement interface.
```

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

The `blink` element is of type `HTMLPhraseElement`, not `HTMLUnknownElement`.

V0371: The isindex element is of type HTMLIsIndexElement, not HTMLUnknownElement

The specification states:

```
11.3.4 Other elements, attributes and APIs
    ...
    The blink, bgsound, isindex, multicol, nextid, and spacer elements must use the
    HTMLUnknownElement interface.
```

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

The `isindex` element is of type `HTMLIsIndexElement`, not `HTMLUnknownElement`.

V0372: The nextid element is of type HTMLNextIdElement, not HTMLUnknownElement

The specification states:

```
11.3.4 Other elements, attributes and APIs
    ...
    The blink, bgsound, isindex, multicol, nextid, and spacer elements must use the
```

```
HTMLUnknownElement interface.
```

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

The `nextid` element is of type `HTMLNextIdElement`, not `HTMLUnknownElement`.

2.2 Clarifications

The following subsections identify clarifications to recommendations made by [HTML5].

2.2.1 [HTML5] Section 2.2.1 Conformance classes

C0001: The developer tools preserve the conformance errors

The specification states:

```
2.2.1 Conformance classes
    ...
    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.
```

All document modes (All versions)

The developer tools preserve the conformance errors and indicate the error 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
    ...
    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.
```

All document modes (All versions)

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

2.2.2 [HTML5] 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
    ...
    Implementations may support WebVTT as a text track format for subtitles, captions,
    chapter titles, metadata, etc, for media resources. [WEBVTT]
```

IE11 mode, IE10 mode, and EdgeHTML Mode (All versions)

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

2.2.3 [HTML5] 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

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.
```

All document modes (All versions)

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 [HTML5] Section 2.6.7 CORS-enabled fetch

C0005: Cross-origin resource access errors are reported to the console

The specification states:

```
2.6.7 CORS-enabled fetch

    When the user agent is required to perform a potentially CORS-enabled fetch of an
    absolute URL URL with a mode mode that is either "No CORS", "Anonymous", or "Use
    Credentials", optionally using a referrer source referrer source, with an origin
    origin, and with a default origin behaviour default which is either "taint" or
    "fail", it must run the first applicable set of steps from the following list.

        If mode is "No CORS" and default is taint
            ...
            ... The user agent may report a cross-origin resource access failure to the
            user (e.g. in a debugging console).

        If mode is "No CORS"
            ...
            ... The user agent may report a cross-origin resource access failure to the
            user (e.g. in a debugging console).
```

IE11 mode, IE10 mode, and EdgeHTML Mode (All versions)

Cross-origin resource access failures are reported to the console.

2.2.5 [HTML5] Section 3.2.5.2 The title attribute

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

The specification states:

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

All document modes (All versions)

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

2.2.6 [HTML5] Section 3.2.5.3 The lang and xml:lang attributes

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

The specification states:

```
3.2.5.3 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).
```

IE11 mode, IE10 mode, IE9 mode, and EdgeHTML Mode (All versions)

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

2.2.7 [HTML5] Section 4.2.4 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 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.
```

All document modes (All versions)

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.

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

The specification states:

```
4.2.4 The 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.
```

All document modes (All versions)

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

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

The specification states:

```
4.2.4 The link element
    ...
    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 sniffed type of the resource
    as if it was the actual type.
```

All document modes (All versions)

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

2.2.8 [HTML5] Section 4.2.5.1 Standard metadata names

C0011: The title, not the application name, is used for UI in cases of page-created dialogs or tabs

The specification states:

```
4.2.5.1 Standard metadata names
    ...
    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.
```

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

The title, not the application name, is used for UI in cases of page-created dialogs or tabs.

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
    ...
```

```
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 (Microsoft Edge)

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

2.2.9 [HTML5] Section 4.2.5.3 Pragma directives

C0012: There is no visual representation of timers or redirects

The specification states:

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

        This pragma acts as timed redirect.

            24. 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.
```

All document modes (All versions)

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

2.2.10 [HTML5] Section 4.3.9 The address element

C0013: The information within an address element is displayed to the user

The specification states:

```
4.3.9 The address element
    ...
    User agents may expose the contact information of a node to the user, or use it for
    other purposes, such as indexing sections based on the sections' contact information.
```

All document modes (All versions)

The information within an `address` element is displayed to the user.

2.2.11 [HTML5] Section 4.4.4 The blockquote element

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

The specification states:

```
4.4.4 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.
```

All document modes (All versions)

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

2.2.12 [HTML5] Section 4.4.7 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.
```

All document modes (All versions)

The maximum value of the `value` attribute is 2,147,483,647. Any `li` element values exceeding that will be clamped to the 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.
```

All document modes (All versions)

The minimum value of the `value` attribute is -2,147,483,648. Any `li` element values exceeding that will be clamped to the minimum.

2.2.13 [HTML5] Section 4.6.3 Attributes common to ins and del elements

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.
```

All document modes (All versions)

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.
```

All document modes (All versions)

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

2.2.14 [HTML5] Section 4.7.1 The img element

C0019: Images are obtained immediately

The specification states:

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

All document modes (All versions)

Images are obtained immediately.

C0020: An unavailable image indicator is shown to the user when no image is available

The specification states:

```
4.7.1 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.
```

IE10 mode, IE9 mode, IE8 mode, IE7 mode, and IE5 (Quirks) mode (All versions)

An unavailable image indicator is shown to the user when no image is available.

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

The specification states:

```
4.7.1 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.
```

IE11 mode and EdgeHTML Mode (All versions)

No image indicator is shown when the image is unavailable. If alt text is available that text will be shown.

2.2.15 [HTML5] Section 4.7.3 The embed element

C0022: The user agent does not provide a way to override the sandbox instantiation

The specification states:

```
4.7.3 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.
```

All document modes (All versions)

The user agent does not provide a way to override the sandbox instantiation.

2.2.16 [HTML5] Section 4.7.6 The video element

C0023: Visual indicators provide the state of the video

The specification states:

```
4.7.6 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.
```

All document modes (All versions)

Visual indicators provide the state of the video.

C0024: No external link is provided if the video cannot be rendered

The specification states:

```
4.7.6 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.
```

All document modes (All versions)

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. full-screen or in an independent resizable window).
```

All document modes (All versions)

Videos can be played fullscreen.

C0026: Fullscreen videos show controls and ignore the controls attribute

The specification states:

```
4.7.6 The video element
    ...
    ... In such an independent context, however, user agents may make full user
    interfaces visible, with, e.g., play, pause, seeking, and volume controls, even if
    the controls attribute is absent.
```

All document modes (All versions)

Fullscreen videos show controls and ignore the `controls` attribute.

C0027: Screensavers are not disabled for fullscreen videos

The specification states:

```
4.7.6 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.
```

All document modes (All versions)

Screensavers are not disabled for fullscreen videos.

2.2.17 [HTML5] Section 4.7.10.5 Loading the media resource

C0028: Users can block downloads for media

The specification states:

```
4.7.10.5 Loading the media resource
    ...
    The resource fetch algorithm for a media element and a given absolute URL is as
    follows:

        4.  Perform a potentially CORS-enabled fetch of the current media resource's
            absolute URL ...

            User agents may allow users to selectively block or slow media data
            downloads. When a media element's download has been blocked altogether, the
            user agent must act as if it was stalled (as opposed to acting as if the
            connection was closed).
```

All document modes (All versions)

Users can block downloads for media.

C0029: There is throttling at the networking level but not at the user agent level

The specification states:

```
4.7.10.5 Loading the media resource
    ...
    The resource fetch algorithm for a media element and a given absolute URL is as
    follows:

        4.  Perform a potentially CORS-enabled fetch of the current media resource's
            absolute URL ...

            ... The rate of the download may also be throttled automatically by the user
            agent, e.g. to balance the download with other connections sharing the same
            bandwidth.
```

All document modes (All versions)

There is throttling at the networking level but not at the user agent level.

C0030: The preload attribute causes preloading of resources

The specification states:

```
4.7.10.5 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.
```

All document modes (All versions)

The `preload` attribute causes preloading of resources.

C0031: Buffered data is discarded only if data becomes invalid

The specification states:

```
4.7.10.5 Loading the media resource
    ...
    User agents may discard previously buffered data.
```

All document modes (All versions)

Buffered data is discarded only if data becomes invalid.

2.2.18 [HTML5] Section 4.7.10.8 Playing the media resource

C0032: Pitch adjustments are made when the playback rate is not 1.0

The specification states:

```
4.7.10.8 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.
```

All document modes (All versions)

Pitch adjustments are made when the playback rate is not 1.0.

2.2.19 [HTML5] Section 4.7.10.12.6 Text tracks describing chapters

C0033: Chapters are not presented to the user in any way

The specification states:

```
4.7.10.12.6 Text tracks describing chapters
```

All document modes (All versions)

Chapters are not presented to the user in any way.

2.2.20 [HTML5] Section 4.7.10.13 User interface

C0034: Controls are not provided if the controls attribute is absent

The specification states:

```
4.7.10.13 User interface
    ...
    Even when the attribute is absent, however, user agents may provide controls to
    affect playback of the media resource (e.g. play, pause, seeking, and volume
    controls), but such features should not interfere with the page's normal rendering.
    ...
```

All document modes (All versions)

Controls are not provided if the `controls` attribute is absent.

C0035: The volume level and mute setting are not retained between navigations

The specification states:

```
4.7.10.13 User interface
    ...
    The volume attribute must return the playback volume of any audio portions of the
    media element, in the range 0.0 (silent) to 1.0 (loudest). Initially, the volume
    should be 1.0, but user agents may remember the last set value across sessions, on a
    per-site basis or otherwise, so the volume may start at other values. ...

    The muted attribute must return true if the audio output is muted and false
    otherwise. Initially, the audio output should not be muted (false), but user agents
    may remember the last set value across sessions, on a per-site basis or otherwise, so
    the muted state may start as muted (true). ...
```

All document modes (All versions)

The volume level and mute setting are not retained between navigations.

2.2.21 [HTML5] Section 4.8 Links

C0036: The processing semantics of hyperlinks are not modified for some link types

The specification states:

```
4.8 Links
    ...
    A hyperlink can have one or more hyperlink annotations that modify the processing
    semantics of that hyperlink.
```

All document modes (All versions)

The processing semantics of hyperlinks are not modified for the following `rel` link types: `author`, `bookmark`, `help`, `license`, `nofollow`, `search` and `tag`.

2.2.22 [HTML5] Section 4.8.1 Links created by a and area elements

C0037: No indicator is provided for the user to choose whether to navigate or download the link

The specification states:

```
4.8.1 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.
```

All document modes (All versions)

No indicator is provided for the user to choose whether to navigate or download the link.

2.2.23 [HTML5] Section 4.10.5.1.5 E-mail state (type=email)

C0038: Invalid email addresses not allowed

The specification states:

```
User agents may allow the user to set the value to a string that is not a valid e-mail
address.
```

All document modes (All versions)

Invalid email addresses are allowed when the `multiple` attribute is defined. Otherwise, invalid email addresses are not allowed.

C0039: Punycode in a value is not properly converted

The specification states:

```
4.10.5.1.5 E-mail state (type=email)
    ...
    How the E-mail state operates depends on whether the multiple attribute is specified
    or not.

        When the multiple attribute is not specified on the element

            The input element represents a control for editing an e-mail address given in
            the element's value.

            ... User agents may transform the values for display and editing; in
            particular, user agents should convert punycode in the value to IDN in the
            display and vice versa. ...
```

All document modes (All versions)

Punycode in a value is not properly converted to IDN.

2.2.24 [HTML5] Section 4.10.5.1.14 File Upload state (type=file)

C0040: The accept attribute is used to filter the file selection from the file picker

The specification states:

```
4.10.5.1.14 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. ...
```

IE11 mode, IE10 mode, IE9 mode, and EdgeHTML Mode (All versions)

The `accept` attribute is used to filter the file selection from the file picker.

2.2.25 [HTML5] Section 4.10.19.3 Limiting user input length: the maxlength attribute

C0041: A negative maxlength value is treated as if it were 0

The specification states:

```
4.10.19.3 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.
```

All document modes (All versions)

A negative `maxlength` value is treated as if it were 0. No characters are accepted.

2.2.26 [HTML5] Section 4.10.19.8 Autofilling form controls: the autocomplete attribute

C0042: When autofilll is not off, control values are stored and previously stored values are offered to the user

The specification states:

```
4.10.19.8 Autofilling form controls: the autocomplete attribute
    ...
    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.
```

IE11 mode, IE10 mode, and EdgeHTML Mode (Microsoft Edge, Internet Explorer 11, and Internet Explorer 10)

When autofilll is not off, control values are stored and previously stored values are offered to the user.

2.2.27 [HTML5] Section 4.10.21.2 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.
```

IE11 mode, IE10 mode, and EdgeHTML Mode (All versions)

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

2.2.28 [HTML5] Section 4.10.22.7 Multipart form data

C0044: Form fields, including filename fields, are encoded in UTF-8 and are not approximated

The specification states:

```
4.10.22.7 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). User agents must not use the RFC 2231 encoding suggested by RFC
            2388.
```

All document modes (All versions)

Form fields, including filename fields, are encoded in UTF-8 and are not approximated.

2.2.29 [HTML5] Section 4.11.4.2 Serializing bitmaps to a file

C0045: Many image formats other than PNG are supported

The specification states:

```
4.11.4.2 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.
```

All document modes (All versions)

Many image formats other than PNG are supported (gif, jpeg, ico, bmp, etc.).

2.2.30 [HTML5] Section 5.1.3 Secondary browsing contexts

C0046: Some dialog boxes are secondary browsing contexts

The specification states:

```
5.1.3 Secondary browsing contexts

    User agents may support secondary browsing contexts, which are browsing contexts that
    form part of the user agent's interface, apart from the main content area.
```

All document modes (All versions)

Some dialog boxes are secondary browsing contexts (for example, the script error dialog).

2.2.31 [HTML5] Section 5.1.6 Browsing context names

C0047: A message is shown to indicate that a popup window was blocked

The specification states:

```
5.1.6 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 algorithm is not allowed to show a popup and the user agent has
                been configured to not show popups (i.e. the user agent has a "popup
```

```
blocker" enabled)

    There is no chosen browsing context. The user agent may inform the
    user that a popup has been blocked.
```

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

A message is shown to indicate that a popup window was blocked.

C0048: A new browsing context is created

The specification states:

```
5.1.6 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.
```

All document modes (All versions)

If the current browsing context's active document's active sandboxing flag set has the sandboxed auxiliary navigation browsing context flag set, a new browsing context is created.

2.2.32 [HTML5] Section 5.2.2 APIs for creating and navigating browsing contexts by name

C0049: The third argument, features, is used to define restrictions on windows that are created

The specification states:

```
5.2.2 APIs for creating and navigating browsing contexts by name
    ...
    The third argument, features, has no defined effect and is mentioned for historical
    reasons only. User agents may interpret this argument as instructions to set the size
    and position of the browsing context, but are encouraged to instead ignore the
    argument entirely.
```

All document modes (All versions)

The third argument, `features`, is used to define restrictions on windows that are created.

2.2.33 [HTML5] Section 5.5.1 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:

```
5.5.1 The session history of browsing contexts
    ...
    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.
```

All document modes (All versions)

Document objects are discarded based on content expiration, disk space usage, and preferences for content storage.

2.2.34 [HTML5] Section 5.5.2 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:

```
5.5.2 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.)
```

IE11 mode, IE10 mode, IE9 mode, and EdgeHTML Mode (All versions)

The maximum number of state objects added to the session history for a page is 1,048,576.

2.2.35 [HTML5] Section 5.5.3 The Location interface

C0052: The user can set a flag to cause caches to be bypassed on a reload

The specification states:

```
5.5.3 The Location interface
    ...
    When a user requests that the active document of a browsing context be reloaded
    through a user interface element, the user agent should navigate the browsing context
    to the same resource as that Document, with replacement enabled. ...
```

IE11 mode, IE10 mode, IE9 mode, and IE8 mode (All versions)

The user can set a flag to cause caches to be bypassed on a reload.

2.2.36 [HTML5] Section 5.6.1 Navigating across documents

C0053: Navigation errors are shown for all document response codes other than code value 200

The specification states:

```
5.6.1 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.
            ...
        18. ...
        19. ...
            ... The user agent may indicate to the user that the original page load
            failed, and that the page used was a previously cached resource.
            ...
        20. ...
            ... but the user agent may indicate to the user that the original page load
            failed, that the page used was a fallback resource, and what the URL of the
            fallback resource actually is.
            ...
```

All document modes (All versions)

Navigation errors are shown for all document response codes other than code value 200.

2.2.37 [HTML5] Section 5.6.3 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:

```
5.6.3 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.
```

All document modes (All versions)

The root element performs a namespace-based lookup in order to determine if the content is a feed.

2.2.38 [HTML5] Section 5.6.4 Page load processing model for text files

C0055: Content is not added to the head element of the document

The specification states:

```
5.6.4 Page load processing model for text files
    ...
    User agents may add content to the head element of the Document, e.g. linking to a
    style sheet or an XBL binding, providing script, giving the document a title, etc.
```

All document modes (All versions)

Content is not added to the head element of the document.

2.2.39 [HTML5] Section 5.6.6 Page load processing model for media

C0056: A head section is added to the content of a Document

The specification states:

```
5.6.6 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 or an XBL binding, to provide a
    script, to give the document a title, to make the media autoplay, etc.
```

All document modes (All versions)

A `head` section is added to the content of a `Document`.

2.2.40 [HTML5] Section 5.6.10 History traversal

C0057: The scroll state is retained for back and forward navigations

The specification states:

```
5.6.10 History traversal
    ...
    When a user agent is required to traverse the history to a specified entry,
    optionally with replacement enabled, and optionally with the 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, for instance the scroll position
            or values of form fields, that it had previously recorded.
```

All document modes (All versions)

The scroll state is retained for back and forward navigations.

2.2.41 [HTML5] Section 5.6.11 Unloading documents

C0058: The standard dialog asking for user confirmation to close the window does not show the returnValue to the user

The specification states:

```
5.6.11 Unloading documents
    ...
    When a user agent is to prompt to unload a document, it must run the following steps.
        ...
        8.  ...
            The prompt shown by the user agent may include the string of the returnValue
            attribute, or some leading subset thereof. (A user agent may want to truncate
            the string to 1024 characters for display, for instance.)
            ...
```

All document modes (All versions)

The standard dialog asking for user confirmation to close the window does not show the `returnValue` to the user.

2.2.42 [HTML5] Section 5.6.12 Aborting a document load

C0059: The user can explicitly invoke the abort a document algorithm by clicking the stop button in the address bar

The specification states:

```
5.6.12 Aborting a document load
    ...
    User agents may allow users to explicitly invoke the abort a document algorithm for a
    Document. ...
```

All document modes (All versions)

The user can explicitly invoke the abort a document algorithm by clicking the stop button in the address bar.

2.2.43 [HTML5] Section 5.7.4 Downloading or updating an application cache

C0060: Caching progress is not shown

The specification states:

```
5.7.4 Downloading or updating an application cache
    ...
    Some of these steps have requirements that only apply if the user agent shows caching
    progress. Support for this is optional. Caching progress UI could consist of a
    progress bar or message panel in the user agent's interface, or an overlay, or
    something else. Certain events fired during the application cache download process
    allow the script to override the display of such an interface. (Such events are
    delayed until after the load event has fired.) The goal of this is to allow Web
    applications to provide more seamless update mechanisms, hiding from the user the
    mechanics of the application cache mechanism. User agents may display user interfaces
    independent of this, but are encouraged to not show prominent update progress
    notifications for applications that cancel the relevant events.
```

All document modes (All versions)

Caching progress is not shown.

2.2.44 [HTML5] Section 5.7.5 The application cache selection algorithm

C0061: The user is not notified of an inconsistency between the cache manifest and the document metadata

The specification states:

```
5.7.5 The application cache selection algorithm
    ...
    When the application cache selection algorithm algorithm is invoked with a Document
    document and optionally a manifest URL manifest URL, the user agent must run the
    first applicable set of steps from the following list:

        If there is a manifest URL, and document was loaded from an application cache,
        and the URL of the manifest of that cache's application cache group is not the
        same as manifest URL
            ...
            User agents may notify the user of the inconsistency between the cache
            manifest and the document's own metadata, to aid in application development.
```

All document modes (All versions)

The user is not notified of an inconsistency between the cache manifest and the document metadata.

2.2.45 [HTML5] Section 5.7.8 Disk space

C0062: Deletion of specific application caches is not supported

The specification states:

```
5.7.8 Disk space
    ...
    User agents should allow users to see how much space each domain is using, and may
    offer the user the ability to delete specific application caches.
```

All document modes (All versions)

Deletion of specific application caches is not supported. The application cache API provides no method to delete specific items.

2.2.46 [HTML5] Section 6.1.2 Enabling and disabling scripting

C0063: The user can set a preference to disable scripting

The specification states:

```
6.1.2 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.)
...
```

All document modes (All versions)

The user can set a preference to disable scripting.

2.2.47 [HTML5] Section 6.1.3.4 Creating scripts

C0064: Script errors are reported to the console or to a popup

The specification states:

```
6.1.3.4 Creating scripts
    ...
    When the specification says that a script is to be created, given some script source,
    a script source URL, its scripting language, a script settings object, and optionally
    a muted errors flag, the user agent must run the following steps:
        ...
        7.  ...
            Otherwise, report the error for script, with the problematic position (line
            number and column number), using the global object specified by the script
            settings object as the target. If the error is still not handled after this,
            then the error may be reported to the user.
```

All document modes (All versions)

Script errors are reported to the console or to a popup.

2.2.48 [HTML5] Section 6.1.3.5 Killing scripts

C0065: The user can kill a script with the stop script feature

The specification states:

```
6.1.3.5 Killing scripts
    ...
    User agents are encouraged to allow users to disable scripting whenever the user is
    prompted either by a script (e.g. using the window.alert() API) or because of a
    script's actions (e.g. because it has exceeded a time limit).
    ...
    User agents may allow users to specifically disable scripts just for the purposes of
    closing a browsing context.
```

IE11 mode, IE10 mode, IE9 mode, and EdgeHTML Mode (All versions)

The user can kill a script with the stop script feature.

C0066: There are no quota restrictions

The specification states:

```
6.1.3.5 Killing scripts

    User agents may impose resource limitations on scripts, for example CPU quotas,
    memory limits, total execution time limits, or bandwidth limitations. ...
```

IE11 mode, IE10 mode, IE9 mode, and EdgeHTML Mode (All versions)

There are no quota restrictions.

2.2.49 [HTML5] Section 6.1.3.6.1 Runtime script errors in documents

C0067: When an option is set, script errors are reported to the user

The specification states:

```
6.1.3.6.1 Runtime script errors in documents

    ... If the error is still not handled after this, then the error may be reported to
    the user.
```

All document modes (All versions)

When the 'Display a notification about every script error' option is set, script errors are reported to the user.

2.2.50 [HTML5] Section 6.1.5.1 Event handlers

C0068: An unparsable body results in an error reported to the user

The specification states:

```
6.1.5.1 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:
            ...
            8.  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
                    to the user.
```

All document modes (All versions)

An unparsable body results in an error reported to the user.

2.2.51 [HTML5] Section 6.5.2 Printing

C0069: Printing events do not wait for the user to accept or decline

The specification states:

```
6.5.2 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.
```

All document modes (All versions)

Printing events do not wait for the user to accept or decline.

2.2.52 [HTML5] Section 7.1 The hidden attribute

C0070: Assistive technologies determine what is done with hidden items

The specification states:

```
7.1 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.
```

All document modes (All versions)

Assistive technologies have access to elements that are in the hidden state, and those technologies determine what is done with the hidden items.

2.2.53 [HTML5] Section 7.2 Inert subtrees

C0071: Tracking focus follows platform conventions

The specification states:

```
7.4 Focus
    ...
    User agents may track focus for each browsing context or Document individually, or
    may support only one focused element per top-level browsing context — user agents
    should follow platform conventions in this regard.
```

All document modes (All versions)

Tracking focus follows platform conventions.

C0072: Selection and find on a page are prevented from working when the page is inert because of a dialog

The specification states:

```
7.2 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. ...
```

All document modes (All versions)

Selection and find on a page are prevented from working when the page is inert because of a dialog.

2.2.54 [HTML5] Section 7.4.2 Focus management

C0073: Focusable elements follow the platform conventions for accessibility

The specification states:

```
7.4.2 Focus management
    ...
    Notwithstanding the above, user agents may make any element or part of an element
    focusable, especially to aid with accessibility or to better match platform
    conventions.
```

All document modes (All versions)

Focusable elements follow the platform conventions for accessibility.

2.2.55 [HTML5] Section 7.4.4 Element-level focus APIs

C0074: The blur function is not ignored on elements but is ignored on the window object

The specification states:

```
7.4.4 Element-level focus APIs
    ...
    The blur() method, when invoked, should run the unfocusing steps for the element on
    which the method was called instead. User agents may selectively or uniformly ignore
    calls to this method for usability reasons.
```

All document modes (All versions)

The `blur` function is not ignored on elements but is ignored on the `window` object.

2.2.56 [HTML5] Section 7.5.3 Processing model

C0075: The fallback assigns an access key which is based on the x-ms-acceleratorKey

The specification states:

```
7.5.3 Processing model
    ...
    Whenever an element's accesskey attribute is set, changed, or removed, the user agent
    must update the element's assigned access key by running the following steps:
        ...
        4.  Fallback: Optionally, the user agent may assign a key combination of its
            choosing as the element's assigned access key and then abort these steps.
```

IE11 mode and IE10 mode (All versions)

The fallback assigns an access key combination which is based on the `x-ms-acceleratorKey`.

C0076: The fallback does not assign an access key combination

The specification states:

```
7.5.3 Processing model
    ...
    Whenever an element's accesskey attribute is set, changed, or removed, the user agent
    must update the element's assigned access key by running the following steps:
        ...
        4.  Fallback: Optionally, the user agent may assign a key combination of its
            choosing as the element's assigned access key and then abort these steps.
```

IE9 mode and IE8 mode (All versions)

The fallback does not assign an access key combination.

2.2.57 [HTML5] Section 7.6.5 Spelling and grammar checking

C0077: The lang attribute defined on an element determines the spellcheck language

The specification states:

```
7.6.5 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.
```

All document modes (All versions)

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:

```
7.6.5 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).
```

All document modes (All versions)

Spelling and grammar errors on the text preloaded with the page are not reported.

2.2.58 [HTML5] 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.
```

All document modes (All versions)

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.

2.2.59 [HTML5] Section 8.2.7 Coercing an HTML DOM into an infoset

C0080: Attributes are dropped if they starts 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.
```

All document modes (All versions)

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.
```

All document modes (All versions)

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 the 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) character, the tool
    may insert a single U+0020 SPACE character at the end of such comments.

    If the XML API restricts allowed characters in character data, attribute values, or
    comments, the tool may replace any "FF" (U+000C) character with a U+0020 SPACE
    character, and any other literal non-XML character with a U+FFFD REPLACEMENT
    CHARACTER.
```

All document modes (All versions)

No space is inserted between consecutive "-" (U+002D) characters, or after one that ends the line.

2.2.60 [HTML5] Section 9.3 Serializing XHTML 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.
```

All document modes (All versions)

When XHTML documents are serialized, prefixes and namespace declarations are adjusted as needed.

2.2.61 [HTML5] 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:

```
11.3.4 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.
```

All document modes (All versions)

The `scheme` attribute is not used as an extension of the `name` attribute.

2.3 Error Handling

There are no additional considerations for error handling.

2.4 Security

There are no additional security considerations.

3 Change Tracking

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

4 Index