[MS-ARIA]:

Internet Explorer Accessible Rich Internet Applications (WAI-ARIA) 1.0 Standards Support Document

---

Revision Summary

Date	Revision History	Revision Class	Comments
9/20/2014	1.0	New	Released new document.
1/22/2015	2.0	Major	Updated for new product version.
7/7/2015	2.1	Minor	Clarified the meaning of the technical content.
11/2/2015	2.2	Minor	Clarified the meaning of the technical content.
12/7/2015	2.3	Minor	Clarified the meaning of the technical content.
3/22/2016	2.4	Minor	Clarified the meaning of the technical content.
7/19/2016	2.5	Minor	Clarified the meaning of the technical content.
7/27/2016	2.6	Minor	Clarified the meaning of the technical content.
11/2/2016	2.6	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 Accessible Rich Internet Applications (WAI-ARIA) 1.0 specification [WAI-ARIA1.0], published 20 March 2014. The browsers provide access to content for assistive technologies though the **Microsoft Active Accessibility (MSAA)** and **Microsoft UI Automation (UIA)** accessibility frameworks.

1.1 Glossary

This document uses the following terms:

Microsoft Active Accessibility (MSAA): A Component Object Model (COM)-based technology that improves the way accessibility aids work with applications running on Microsoft Windows. It provides dynamic-link libraries that are incorporated into the operating system as well as a COM interface and API elements that provide reliable methods for exposing information about UI elements.

Microsoft UI Automation (UIA): The accessibility model for Microsoft Windows that programmatically gathers information about an application's User Interface (UI) elements and exposes it to assistive technology products and automated test scripts. UI Automation is the successor to the Microsoft Active Accessibility (MSAA) framework.

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

1.2 References

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

1.2.1 Normative References

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

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

[WAI-ARIA1.0] Craig, J., and Cooper, M., Eds., "Accessible Rich Internet Applications (WAI-ARIA) 1.0", W3C Recommendation, March 2014, http://www.w3.org/TR/2014/REC-wai-aria-20140320/

1.2.2 Informative References

None.

1.3 Microsoft Implementations

The following Microsoft web browser versions implement some portion of the [WAI-ARIA1.0] specification:

- Windows Internet Explorer 8
- 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 in each browser version.

Browser Version	Document Modes Supported
Internet Explorer 8	Quirks Mode IE7 Mode IE8 Mode
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:

IE8 Mode, IE9 Mode, and IE10 Mode (All Versions)

1.4 Standards Support Requirements

To conform to [WAI-ARIA1.0], 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 [WAI-ARIA1.0] and whether they are considered normative or informative.

Sections	Normative/Informative
1, 2	Informative
3	Normative
4	Informative
5-9	Normative
10	Informative

1.5 Notation

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

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

For document mode and browser version notation, see also section 1.3.

2 Standards Support Statements

This section contains all variations and clarifications for the Microsoft implementation of [WAI-ARIA1.0].

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

2.1 Normative Variations

The following subsections describe normative variations from the MUST requirements of [WAI-ARIA1.0].

2.1.1 [WAI-ARIA1.0] Section 5.2.4 Inherited States and Properties

V0001:

The specification states:

> States and properties are inherited from superclass roles in the role taxonomy, not from ancestor elements in the DOM tree. These properties are not explicitly defined on the role, as the inheritance of properties is automatic.

IE10 Mode, IE11 Mode, and EdgeHTML Mode (All Versions)

`aria-readonly` is not inherited.

2.1.2 [WAI-ARIA1.0] Section 5.2.7.3, Text Alternative Computation

V0002:

The specification states:

> The text equivalent computation outlined below is a description of how user agents acquire a name or description that they then publish through the accessibility API.

IE10 Mode and IE11 Mode (All Versions)

The text alternative computation algorithm is not supported.

2.1.3 [WAI-ARIA1.0] Section 5.2.8, Presentational Children

V0003:

The specification states:

> The DOM descendants are presentational. User agents SHOULD NOT expose descendants of this element through the platform accessibility API. If user agents do not hide the descendant nodes, some information may be read twice.

IE10 Mode and IE11 Mode (All Versions)

The `img`, `progress`, `separator`, and `range` roles expose their child nodes to the accessibility tree, despite being defined in the spec as having presentational children.

2.1.4 [WAI-ARIA1.0] Section 5.3.4, Landmark Roles

V0004:

The specification states:

> The following roles are regions of the page intended as navigational landmarks. All of these roles inherit from the landmark base type and, with the exception of application, all are imported from the Role Attribute [ROLE]. The roles are included here in order to make them clearly part of the WAI-ARIA Role taxonomy.
>
> - application
> - banner
> - complementary
> - contentinfo
> - form
> - main
> - navigation
> - search

IE8 Mode, IE9 Mode, IE10 Mode, and IE11 Mode (All Versions)

Landmark roles are not supported, however the `application` role is supported for IE8, IE9, IE10 and IE11.

2.1.5 [WAI-ARIA1.0] Section 5.4, Definition of Roles

V0005:

The specification states:

```
article
  A section of a page that consists of a composition that forms an independent part of a
  document, page, or site.
  ...
definition
  A definition of a term or concept.
  ...
log
  A type of live region where new information is added in meaningful order and old
  information may disappear. See related marquee.
  ...
math
  Content that represents a mathematical expression.
  ...
note
  A section whose content is parenthetic or ancillary to the main content of the resource.
  ...
row
  A row of cells in a grid.
  ...
rowgroup
  A group containing one or more row elements in a grid.
  ...
scrollbar
  A graphical object that controls the scrolling of content within a viewing area,
  regardless of whether the content is fully displayed within the viewing area.
```

```
  ...
timer
  A type of live region containing a numerical counter which indicates an amount of elapsed
  time from a start point, or the time remaining until an end point.
```

IE8 Mode and IE9 Mode (All Versions)

Not supported.

2.1.6 [WAI-ARIA1.0] Section 6.5.4, Relationship Attributes

V0006:

The specification states:

```
This section lists attributes that indicate relationships or associations between elements
which cannot be readily determined from the document structure.

  • aria-activedescendant
  • aria-controls
  • aria-describedby
  • aria-flowto
  • aria-labelledby
  • aria-owns
  • aria-posinset
  • aria-setsize
```

IE10 Mode and IE11 Mode (All Versions)

`aria-describedby`, `aria-labelledby`, `aria-flowto`, and `aria-owns` are not supported when the referenced object is not accessible.

2.1.7 [WAI-ARIA1.0] Section 6.6, Definitions of States and Properties (all aria-* attributes)

V0007:

The specification states:

```
aria-atomic
  Indicates whether assistive technologies will present all, or only parts of, the changed
  region based on the change notifications defined by the aria-relevant attribute. See
  related aria-relevant.
  ...
aria-autocomplete
  Indicates whether user input completion suggestions are provided.
  ...
aria-dropeffect
  Indicates what functions can be performed when the dragged object is released on the drop
  target. This allows assistive technologies to convey the possible drag options available
  to users, including whether a pop-up menu of choices is provided by the application.
  Typically, drop effect functions can only be provided once an object has been grabbed for
  a drag operation as the drop effect functions available are dependent on the object being
  dragged.
  ...
aria-grabbed (state)
  Indicates an element's "grabbed" state in a drag-and-drop operation.
  ...
aria-label
  Defines a string value that labels the current element. See related aria-labelledby.
  ...
```

```
aria-multiline
  Indicates whether a text box accepts multiple lines of input or only a single line.
  ...
aria-orientation
  Indicates whether the element and orientation is horizontal or vertical.
  ...
aria-sort
  Indicates if items in a table or grid are sorted in ascending or descending order.
  ...
aria-valuetext
  Defines the human readable text alternative of aria-valuenow for a range widget.
```

IE8 Mode and IE9 Mode (All Versions)

Not supported.

IE10 Mode and IE11 Mode (All Versions)

`aria-dropeffect` and `aria-grabbed` are not supported.

V0009:

The specification states:

```
Values of aria-checked


True
  The element is checked

false
  The element supports being checked but is not currently checked.

mixed
  Indicates a mixed mode value for a tri-state checkbox or menuitemcheckbox.

undefined (default)
  The element does not support being checked
```

IE10 Mode and IE11 Mode (All Versions)

The `mixed` value is not supported.

V0010:

The specification states:

```
aria-activedescendant (property)
  Identifies the currently active descendant of a composite widget.
```

IE10 Mode, IE11 Mode, and EdgeHTML Mode (All Versions)

The `aria-activedescendant` property is only partially supported.

V0011:

The specification states:

```
aria-selected (state)
```

```
Indicates the current "selected" state of various widgets. See related aria-checked and
aria-pressed.

This attribute is used with single-selection and multiple-selection widgets:

  1. Single-selection containers where the currently focused item is not selected. The
     selection normally follows the focus, and is managed by the user agent.

  2. Multiple-selection containers. Authors SHOULD ensure that any selectable descendant
     of a container in which the aria-multiselectable attribute is true specifies a value
     of either true or false for the aria-selected attribute.
```

IE10 Mode, IE11 Mode, and EdgeHTML Mode (All Versions)

The UIA `SelectionPattern` control pattern will function only if a widget has selectable children (as indicated by an `aria-selected` value of `true` or `false`). Thus the UIA `SelectionItemPattern` will not function if `aria-selected` has a value of `undefined` (the default value).

V0012:

The specification states:

```
Characteristics of aria-selected

  Used in Roles:
    gridcell
    option
    row
    tab

  Inherits into Roles:
    columnheader
    menuitemradio
    radio
    rowheader
    treeitem
```

IE10 Mode and IE11 Mode (All Versions)

The existence of a specified `aria-selected` or `aria-checked` attribute (with `true` or `false` value) is used to determine if an element is selectable or not, rather than the [WAI-ARIA1.0] role inheritance model as specified. Thus in some situations the resulting role behavior may conform to that implied by the specification, and in other situations it may differ.

V0013:

The specification states:

```
aria-atomic
  Indicates whether assistive technologies will present all, or only parts of, the changed
  region based on the change notifications defined by the aria-relevant attribute. See
  related aria-relevant.
  ...
aria-haspopup
  Indicates that the element has a popup context menu or sub-level menu.
  ...
aria-relevant
  Indicates what user agent change notifications (additions, removals, etc.) assistive
  technologies will receive within a live region. See related aria-atomic.
```

EdgeHTML Mode (All Versions)

Not supported.

2.2 Clarifications

The following subsections describe clarifications of the MAY and SHOULD requirements of [WAI-ARIA1.0].

2.2.1 [WAI-ARIA1.0] Section 5.4, Definition of Roles

C0001:

The specification states:

```
presentation (role)
  ...
  For any element with a role of presentation and which is not focusable, the user agent
  MUST NOT expose the implicit native semantics of the element (the role and its states and
  properties) to accessibility APIs. However, the user agent MUST expose content and
  descendant elements that do not have an explicit or inherited role of presentation. Thus,
  the presentation role causes a given element to be treated as having no role or to be
  removed from the accessibility tree, but does not cause the content contained within the
  element to be removed from the accessibility tree.
```

IE10 Mode, IE11 Mode, and EdgeHTML Mode (All Versions)

Elements assigned to the presentation role are removed from the accessibility tree; however, they are not placed back into the tree when other accessibility properties (that would normally override the presentation role) are set.

C0002:

The specification states:

```
Characteristics of menuitemradio
  ...
  Inherited States and Properties:
    ...
    aria-checked (state) (required)
...
Characteristics of radio
  ...
  Inherited States and Properties:
    ...
    aria-checked (state) (required)
```

IE10 Mode and IE11 Mode (All Versions)

The `aria-checked` state is supported for `menuitemradio` and `radio` roles, but unlike for other roles, it is mapped to the *selected* state value (`STATE_SYSTEM_SELECTED` for MSAA, `IsSelected` for UIA) instead of the *checked* state value (`STATE_SYSTEM_CHECKED` for MSAA, `ToggleState` for UIA).

2.2.2 [WAI-ARIA1.0] Section 6.6, Definitions of States and Properties (all aria-* attributes)

C0003:

The specification states:

```
aria-busy (state)
  Indicates whether an element, and its subtree, are currently being updated.
  ...
aria-haspopup (property)
  Indicates that the element has a popup context menu or sub-level menu.
  ...
aria-sort
  Indicates if items in a table or grid are sorted in ascending or descending order.
```

IE10 Mode and IE11 Mode (All Versions)

`aria-sort` is only supported in UIA (with the equivalent property of `UIA_ItemStatusPropertyId`), and not in MSAA.

2.3 Error Handling

There are no additional error handling considerations.

2.4 Security

There are no additional security considerations.

3 Change Tracking

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

4 Index