
# ARIA Widget Checklist:
# For Screen Reader Testing

Bryan Garaventa
2/23/17

## Introduction

One of the most significant challenges with ARIA support is determining support level differences between assistive technology and browser combinations, and doing so in a structured manner so that bugs can be accurately and reliably submitted to AT venders so that these support levels can steadily increase equally.

However there are still significant discrepancies between valid ARIA usage and what is supported between specific screen readers and browser combinations, so there is immense benefit for such screen reader venders to get on the same page and test these assistive technologies more thoroughly and to also involve the public for crowd sourcing this feedback for entering more bugs related to these issues.

To aid in this effort the following ARIA widget checklist has been compiled, which can be used to validate screen reader behavior and to determine where support levels still need to be improved. All are coded with strict adherence to the ARIA specification and cover the most common ARIA implementation usages. This is not meant to cover all possible element, role, and attribute combinations, but rather, to test the screen reader support of specific interactive widget roles when applied according to spec. This will also allow for support within any valid element combination if properly supported by the browser and assistive technology.

This is not a duplication of the W3C ARIA APG that is currently being worked on, but rather, is meant to be a simple single-page checklist for testing screen reader support for the most commonly used interactive ARIA widgets when coded properly according to the ARIA specification.

A redistributable PDF download of this information is available here for easy reference. For mobile device testing, the live blog post of this article may be used, or all of these accessible widgets may be downloaded from GitHub.

For sighted developers, Visual ARIA may be used to expose the ARIA role and supporting attribute usages on all of these ARIA implementations.

*Screen Reader Testing Checklist*

### 1. Simulated Checkbox controls

Includes role="checkbox" and aria-checked.

Expected behavior: It should be possible to focus to the Checkbox in the same manner as a standard HTML input+type="checkbox" element and use the Tab and Shift+Tab keystrokes to navigate between them,
when the Checkbox receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced,

and the role="checkbox" checked state should be reliably conveyed when aria-checked is toggled between "false" or "mixed" or "true" when applicable.

Perform Test

Relevant Mappings: checkbox

2. Simulated Button and Link controls

Includes role="button" and role="link".

Expected behavior: It should be possible to focus to the Button and Link in the same manner as a standard HTML button or link element and use the Tab and Shift+Tab keystrokes to navigate between them,
and when the Button or Link receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced.

Perform Test

Relevant Mappings: button, link

3. ARIA Toggle buttons

Includes role="button" and aria-pressed.

Expected behavior: It should be possible to focus to the Toggle Buttons in the same manner as a standard HTML button element and use the Tab and Shift+Tab keystrokes to navigate between them,
when the Toggle Button receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced,
and the role="button" pressed state should be reliably conveyed when aria-pressed is toggled between "false" or "true".

Perform Test:

Relevant Mappings: button

4. Toggle Button Accordion controls

Includes role="heading", aria-level, role="button", aria-pressed, aria-expanded, aria-controls, role="region", and aria-labelledby.

Expected behavior: It should be possible to focus to the Accordion Toggle Buttons in the same manner as a standard HTML button element and use the Tab and Shift+Tab keystrokes to navigate between them,

when the Toggle Button receives focus both the explicit Name via aria-labelledby or aria-label or embedded content and the pressed and expanded state should be announced,
the role="button" pressed state should be reliably conveyed when aria-pressed is toggled between "false" or "true",
the role="button" expanded state should be reliably conveyed when aria-expanded is toggled between "false" or "true",
and the associated named Region should be conveyed when exposed.

Perform Test

Relevant Mappings: button, heading, region

5. Expandable Link Accordion controls

Includes role="heading", aria-level, native links, aria-expanded, aria-controls, role="region", and aria-labelledby.

Expected behavior: It should be possible to focus to the Accordion Expandable Links in the same manner as a standard HTML link element and use the Tab and Shift+Tab keystrokes to navigate between them,
when the Expandable Link receives focus both the explicit Name via aria-labelledby or aria-label or embedded content and the expanded state should be announced,
the Link expanded state should be reliably conveyed when aria-expanded is toggled between "false" or "true",
and the associated named Region should be conveyed when exposed.

Perform Test

Relevant Mappings: link, heading, region

6. Basic simulated Radio controls

Includes role="radiogroup", role="radio", aria-posinset, aria-setsize, aria-checked, and aria-owns.

Expected behavior: It should be possible to focus to the Radio in the same manner as a standard HTML input+type="radio" element and use the arrow keys to navigate between them,
when the Radio receives focus both the explicit Name via aria-labelledby or aria-label or embedded content and the current checked state should be announced,
the role="radio" checked state should be reliably conveyed when aria-checked is toggled between "false" or "true",
X of Y positioning should be conveyed via aria-posinset and aria-setsize,
and the shared Radiogroup label should be announced when focus is set to the role="radio" element.

Perform Test

Relevant Mappings: radiogroup

7. Horizontal Slider control

Includes role="slider", aria-label, aria-valuemin, aria-valuemax, aria-valuenow, aria-valuetext, and aria-orientation.

Expected behavior: It should be possible to focus to the Slider in the same manner as a standard form field and use the arrow keys to change the value of the Slider,
instructions for pressing the Left/Right arrow keys to navigate a 'horizontal' slider should be conveyed when the Slider receives focus to reflect the value of aria-orientation="horizontal" or if aria-orientation is missing,
when the Slider receives focus both the explicit Name via aria-labelledby or aria-label or embedded content and the current value of the Slider should be announced,
and as the arrow keys change the Slider value, this change should automatically be announced as set within aria-valuetext or within aria-valuenow if aria-valuetext is missing.

Perform Test

Relevant Mappings: slider

8. Vertical Slider control

Includes role="slider", aria-label, aria-valuemin, aria-valuemax, aria-valuenow, aria-valuetext, and aria-orientation.

Expected behavior: It should be possible to focus to the Slider in the same manner as a standard form field and use the arrow keys to change the value of the Slider,
instructions for pressing the Up/Down arrow keys to navigate a 'vertical' slider should be conveyed when the Slider receives focus to reflect the value of aria-orientation="vertical",
when the Slider receives focus both the explicit Name via aria-labelledby or aria-label or embedded content and the current value of the Slider should be announced,
as the arrow keys change the Slider value, this change should automatically be announced as set within aria-valuetext or within aria-valuenow if aria-valuetext is missing.

Perform Test

Relevant Mappings: slider

9. Single-select simulated Listbox control

Includes role="listbox", role="option", aria-posinset, aria-setsize, aria-selected, aria-label, and aria-owns.

Expected behavior: It should be possible to focus to the Listbox in the same manner as a standard HTML select element and use the Up/Down arrow keys to navigate between each role="option" element,
when the Listbox receives focus the explicit Name via aria-labelledby or aria-label should be announced,
when Up/Down is used to move between each role="option" element the explicit Name of each Option should be announced,
and the X of Y positioning should be conveyed via aria-posinset and aria-setsize.

Perform Test

Relevant Mappings: listbox

10. Single-select simulated Listbox control plus tri-state Checkable Options

Includes role="listbox", role="option", aria-checked, aria-posinset, aria-setsize, aria-selected, aria-label, and aria-owns.

Expected behavior: It should be possible to focus to the Listbox in the same manner as a standard HTML select element and use the Up/Down arrow keys to navigate between each role="option" element,
when the Listbox receives focus the explicit Name via aria-labelledby or aria-label should be announced,
arrowing Up/Down should announce both the focused role="option" Name and the checked state via aria-checked,
pressing the Spacebar or clicking or tapping a Listbox role="option" element should be conveyed as toggling checkability to reflect the current value of aria-checked,
and the X of Y positioning should be conveyed via aria-posinset and aria-setsize.

Perform Test

Relevant Mappings: listbox

11. Multiselectable simulated Listbox control

Includes role="listbox", role="option", aria-posinset, aria-setsize, aria-selected, aria-label, aria-multiselectable, and aria-owns.

Expected behavior: It should be possible to focus on the Listbox and enter Applications Mode in the same manner as a standard HTML select element and use the Up/Down arrow keys to navigate between each role="option" element,
focusing on the Listbox should announce both the Listbox Name and the focused role="option" Name,

arrowing Up/Down should announce both the focused role="option" Name and the select state via aria-selected,
pressing the Spacebar or clicking or tapping a Listbox role="option" element should be conveyed as toggling selection to reflect the current value of aria-selected,
and the X of Y positioning should be conveyed via aria-posinset and aria-setsize.

Perform Test

Relevant Mappings: listbox

12. Substring Editable Combobox

Includes role="combobox", aria-activedescendant, aria-expanded, aria-controls, aria-autocomplete, and aria-describedby (plus the role="listbox" design pattern).

Expected behavior: It should be possible to focus to the editable Combobox in the same manner as a standard HTML input+type="text" element and use the Up/Down arrow keys to navigate between each Listbox role="option" element as referenced via aria-activedescendant,
when the Combobox receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced,
when Up/Down is used to move between each Listbox role="option" element by setting aria-activedescendant the explicit Name of each Option should be announced,
and when typing into the edit field the first search result should be announced via aria-live="polite".

Perform Test

Relevant Mappings: combobox, aria-live

13. Simulated readonly Combobox

Includes role="combobox", aria-activedescendant, aria-expanded, aria-controls, aria-autocomplete, and aria-describedby (plus the role="listbox" design pattern).

Expected behavior: It should be possible to focus to the readonly Combobox in the same manner as a standard HTML select element and use the Up/Down arrow keys to navigate between each Listbox role="option" element as referenced via aria-activedescendant,
when the Combobox receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced,
and when Up/Down is used to move between each Listbox role="option" element by setting aria-activedescendant the explicit Name of each Option should be announced.

Perform Test

Relevant Mappings: combobox

14. Interactive Grid control

Includes role="grid", role="rowgroup", role="row", role="gridcell", role="rowheader", role="columnheader", aria-describedby, aria-selected, aria-owns, and aria-labelledby.

Expected behavior: In Virtual Cursor mode or equivalent if supported table navigation commands should move focus between each cell/row like a data table,
setting focus to the Grid in Applications Mode if supported should move focus between each cell/row when the Up/Right/Down/Left arrow keys are pressed,
as focus moves between each cell the associated column and row header cell Names should be announced as set using aria-describedby,
and when aria-selected is set on either the row or cell this should be announced in both Virtual Cursor and Applications Mode when applicable.

Perform Test

Relevant Mappings: grid

15. Disabled date range compound component date picker control

Includes role="application", role="dialog", role="button", role="link", aria-label, and aria-disabled.

Expected behavior: When focus is set within the datepicker after rendering screen readers should enforce Applications Mode via role="application" to ensure that the Up/Right/Down/Left/Home/End/PageUp/PageDown keystrokes are passed through directly to the element that has focus,
and when each date Link receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced.

Perform Test

Relevant Mappings: application, dialog, button, link

16. Button: Basic left-click horizontal popup Menu control

Includes role="application", role="menubar", role="menuitem", aria-posinset, aria-setsize, aria-haspopup, and aria-owns.

Expected behavior: Activating the Menu Button should cause the Menubar to render and focus be set upon the first role="menuitem" element,
instructions for pressing the Left/Right arrow keys to navigate a 'horizontal' menu should be conveyed due to the first level parent of role="menubar",

as the arrow keys are pressed to move focus between each role="menuitem" element the Name of the Menuitem should be conveyed,
when focus is set upon a role="menuitem" element that includes aria-haspopup="true" the presence of a submenu should be announced,
the state of aria-expanded should be announced to convey when a role="menuitem" element has focus that also opens a submenu via aria-haspopup="true",
and Applications Mode should be automatically invoked to allow each key press to be passed through to the Menuitem that currently has focus.

Perform Test

Relevant Mappings: button, application, menubar

17. Button: Basic left-click vertical popup Menu control

Includes role="application", role="menu", role="menuitem", aria-posinset, aria-setsize, aria-haspopup, and aria-owns.

Expected behavior: Activating the Menu Button should cause the Menu to render and focus be set upon the first role="menuitem" element,
instructions for pressing the Up/Down arrow keys to navigate a 'vertical' menu should be conveyed due to the first level parent of role="menu",
as the arrow keys are pressed to move focus between each role="menuitem" element the Name of the Menuitem should be conveyed,
when focus is set upon a role="menuitem" element that includes aria-haspopup="true" the presence of a submenu should be announced,
the state of aria-expanded should be announced to convey when a role="menuitem" element has focus that also opens a submenu via aria-haspopup="true",
and Applications Mode should be automatically invoked to allow each key press to be passed through to the Menuitem that currently has focus.

Perform Test

Relevant Mappings: button, application, menu

18. ARIA Textbox: Basic right-click Popup Menu control

Includes role="textbox", role="application", role="menu", role="menuitem", aria-posinset, aria-setsize, aria-haspopup, and aria-owns.

Expected behavior: It should be possible to type directly into the simulated role="textbox" field in the same manner as a standard HTML textarea element,
pressing Shift+F10 or the Applications key or right-clicking or equivalent on touch devices upon the edit field should open the ARIA vertical Menu construct and move focus to the first role="menuitem" element,

instructions for pressing the Up/Down arrow keys to navigate a 'vertical' menu should be conveyed due to the first level parent of role="menu",
as the arrow keys are pressed to move focus between each role="menuitem" element the Name of the Menuitem should be conveyed,
when focus is set upon a role="menuitem" element that includes aria-haspopup="true" the presence of a submenu should be announced,
the state of aria-expanded should be announced to convey when a role="menuitem" element has focus that also opens a submenu via aria-haspopup="true",
and Applications Mode should be automatically invoked to allow each key press to be passed through to the Menuitem that currently has focus.

Perform Test

Relevant Mappings: textbox, application, menu

19. ARIA Tab group

Includes role="tablist", role="tab", aria-posinset, aria-setsize, aria-expanded, aria-selected, aria-controls, role="tabpanel", aria-labelledby, and aria-owns.

Expected behavior: It should be possible to focus to the Tab in the same manner as a standard HTML input+type="radio" element and use the arrow keys to navigate between them,
when the Tab receives focus both the explicit Name via aria-labelledby or aria-label or embedded content and the current selected and expanded state should be announced when applicable,
the role="tab" selected state should be reliably conveyed when aria-selected is toggled between "false" or "true",
the role="tab" expanded state should be reliably conveyed when aria-expanded is toggled between "false" or "true",
X of Y positioning should be conveyed via aria-posinset and aria-setsize,
and the associated Tabpanel region should be conveyed when exposed.

Perform Test

Relevant Mappings: tablist

20. ARIA Tab group: Complex with right-click Popup Menu control

Includes role="tablist", role="tab", aria-posinset, aria-setsize, aria-expanded, aria-selected, aria-controls, role="tabpanel", aria-labelledby, and aria-owns (Plus the ARIA Menu design pattern).

Expected behavior: It should be possible to focus to the Tab in the same manner as a standard HTML input+type="radio" element and use the arrow keys to navigate between them,

when the Tab receives focus both the explicit Name via aria-labelledby or aria-label or embedded content and the current selected and expanded state should be announced when applicable,
the role="tab" selected state should be reliably conveyed when aria-selected is toggled between "false" or "true",
the role="tab" expanded state should be reliably conveyed when aria-expanded is toggled between "false" or "true",
X of Y positioning should be conveyed via aria-posinset and aria-setsize,
the associated Tabpanel region should be conveyed when exposed,
pressing Shift+F10 or the Applications key or right-clicking or equivalent on touch devices upon the edit field should open the ARIA vertical Menu construct and move focus to the first role="menuitem" element,
instructions for pressing the Up/Down arrow keys to navigate a 'vertical' menu should be conveyed due to the first level parent of role="menu",
as the arrow keys are pressed to move focus between each role="menuitem" element the Name of the Menuitem should be conveyed,
when focus is set upon a role="menuitem" element that includes aria-haspopup="true" the presence of a submenu should be announced,
the state of aria-expanded should be announced to convey when a role="menuitem" element has focus that also opens a submenu via aria-haspopup="true",
and Applications Mode should be automatically invoked to allow each key press to be passed through to the Menuitem that currently has focus.

Perform Test

Relevant Mappings: tablist, application, menu

21. ARIA Tree control

Includes role="tree", role="group", aria-label, role="treeitem", aria-expanded, aria-selected, aria-level, aria-posinset, aria-setsize, and aria-owns.

Expected behavior: It should be possible to focus to the Tree in the same manner as a standard HTML select element and use the arrow keys to navigate between each role="treeitem" element,
when the Tree receives focus both the explicit Name via aria-labelledby or aria-label and the current Treeitem Name plus selected and expanded state if applicable should be announced,
the role="treeitem" selected state should be reliably conveyed when aria-selected is toggled between "false" or "true",
the role="treeitem" expanded state should be reliably conveyed when aria-expanded is toggled between "false" or "true" on branch nodes,
the role="treeitem" level should be reliably conveyed via aria-level when applicable,
and the X of Y positioning should be conveyed via aria-posinset and aria-setsize.

Perform Test

Relevant Mappings: tree

22. ARIA Tree: Complex with right-click Popup Menu control

Includes role="tree", role="group", aria-label, role="treeitem", aria-expanded, aria-selected, aria-level, aria-posinset, aria-setsize, and aria-owns (Plus the ARIA Menu design pattern).

Expected behavior: It should be possible to focus to the Tree in the same manner as a standard HTML select element and use the arrow keys to navigate between each role="treeitem" element,
when the Tree receives focus both the explicit Name via aria-labelledby or aria-label and the current Treeitem Name plus selected and expanded state if applicable should be announced,
the role="treeitem" selected state should be reliably conveyed when aria-selected is toggled between "false" or "true",
the role="treeitem" expanded state should be reliably conveyed when aria-expanded is toggled between "false" or "true" on branch nodes,
the role="treeitem" level should be reliably conveyed via aria-level when applicable,
X of Y positioning should be conveyed via aria-posinset and aria-setsize,
pressing Shift+F10 or the Applications key or right-clicking or equivalent on touch devices upon the edit field should open the ARIA vertical Menu construct and move focus to the first role="menuitem" element,
instructions for pressing the Up/Down arrow keys to navigate a 'vertical' menu should be conveyed due to the first level parent of role="menu",
as the arrow keys are pressed to move focus between each role="menuitem" element the Name of the Menuitem should be conveyed,
when focus is set upon a role="menuitem" element that includes aria-haspopup="true" the presence of a submenu should be announced,
the state of aria-expanded should be announced to convey when a role="menuitem" element has focus that also opens a submenu via aria-haspopup="true",
and Applications Mode should be automatically invoked to allow each key press to be passed through to the Menuitem that currently has focus.

Perform Test

Relevant Mappings: tree, application, menu

23. Customized auto-rotating Carousel control

Includes role="region" and aria-live="polite".

Expected behavior: It should be possible to focus to the carousel Buttons in the same manner as a standard HTML button element and use the Tab and Shift+Tab keystrokes to navigate between them,
when the Button receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced,

the associated named Region should be conveyed when exposed,
and when a carousel button is activated the new slide content should be announced via aria-live="polite".

Perform Test

Relevant Mappings: region, aria-live

24. Stationary Slideshow control

Includes role="region" and aria-live="polite".

Expected behavior: It should be possible to focus to the slideshow Buttons in the same manner as a standard HTML button element and use the Tab and Shift+Tab keystrokes to navigate between them,
when the Button receives focus the explicit Name via aria-labelledby or aria-label or embedded content should be announced,
the associated named Region should be conveyed when exposed,
and when a slideshow button is activated the new slide content should be announced via aria-live="polite".

Perform Test

Relevant Mappings: region, aria-live

25. Simple Help Tooltip control

Includes role="region" and aria-live="polite".

Expected behavior: Setting focus to and typing into the edit field should cause a dynamic tooltip to appear,
the tooltip text should automatically be announced when rendered via aria-live="polite",
continuing to type into the edit field will cause the tooltip to automatically change which should also be announced via aria-live="polite",
and moving focus away from the edit field should cause the tooltip to disappear.

Perform Test

Relevant Mappings: region, aria-live

26. Simple Error Tooltip control

Includes role="region", aria-required, and aria-live="polite".

Expected behavior: Setting focus to a required field and then pressing Tab to move away from the empty field should cause a dynamic error tooltip to appear,
the error tooltip text should automatically be announced when rendered via aria-live="polite",
and typing into and moving focus away from the edit field should cause the tooltip to disappear.

Perform Test

Relevant Mappings: region, aria-live

27. Modal Dialog control

Includes role="dialog", aria-label, aria-describedby, aria-modal, and aria-hidden.

Expected behavior: Activating the Login button should generate a modal dialog,
focus should be set to the Username field when rendered and the dialog Name and Description should be announced,
the background content should be hidden from screen reader users,
pressing Tab and Shift+Tab should move focus only between focusable active elements within the dialog,
and when the dialog is closed focus should be set back to the Login button.

Perform Test

Relevant Mappings: dialog

28. Agressive Modal Dialog control

Includes role="dialog", role="alert", aria-label, aria-describedby, aria-modal, and aria-hidden.

Expected behavior: Activating the Login button should generate a modal dialog,
focus should be set to the Username field when rendered and the dialog Name and Description should be announced by firing a system alert,
the background content should be hidden from screen reader users,
pressing Tab and Shift+Tab should move focus only between focusable active elements within the dialog,
and when the dialog is closed focus should be set back to the Login button.

Perform Test

Relevant Mappings: dialog, alert

29. Basic Popup control

Includes role="region" and aria-live="polite".

Expected behavior: The triggering element expanded state should be reliably conveyed when aria-expanded is toggled between "false" or "true",
activating the triggering element should generate a popup overlay,
the popup text should automatically be announced when rendered via aria-live="polite",
focus should be set to the beginning of the popup content when rendered,
the associated named Region should be conveyed when exposed,
and when the popup is closed focus should be set back to the triggering element.

Perform Test

Relevant Mappings: region, aria-live

30. Popup Progressbar control

Includes role="progressbar", aria-valuemin, aria-valuemax, aria-valuenow, and aria-valuetext.

Expected behavior: Activating the Download button should render a popup progress bar control,
focus should not be forced to any location on the page when the progress bar is rendered,
the associated named Region should be conveyed when exposed,
and as the progress bar value changes, this change should automatically be announced as set within aria-valuetext or within aria-valuenow if aria-valuetext is missing.

Perform Test

Relevant Mappings: region, progressbar

31. ARIA Tooltips

Includes role="region", role="tooltip", and aria-describedby.

Expected behavior: Setting focus to or mousing over the image link or Password field should cause a dynamic tooltip to appear,
the tooltip text should automatically be announced when rendered via aria-describedby (thus triggering a description_changed event),
and moving focus away or mousing out of the image link or Password field should cause the tooltip to disappear.

Perform Test

Relevant Mappings: region, tooltip

32. Simple Web Chat control

Includes role="region", role="application", and aria-live="polite".

Expected behavior: Activating the Chat button will open a chat message region,
focus should automatically be set on the chat message edit field when rendered,
when the edit field receives focus, screen readers should enforce Applications Mode via role="application" to ensure that the Enter keystroke is passed through directly to the element that has focus,
the associated named Region should be conveyed when exposed,
and the tooltip and incoming message text should automatically be announced when rendered via aria-live="polite".

Perform Test

Relevant Mappings: region, application, aria-live

33. Simple Checkbox controls

Includes aria-live="polite".

Expected behavior: When a checkbox is toggled the status message should automatically be announced when rendered via aria-live="polite".

Perform Test

Relevant Mappings: aria-live

Contact:

Bryan Garaventa
Accessibility Fellow
SSB BART Group, Inc.
bryan.garaventa@ssbbartgroup.com
415.624.2709 (o)