

Keyboard Access in Sims

Related Documents:

- [Complex Controls and Interactions](#)

[1. Keyboard Accessibility: An Introduction](#)

[2. Process for Designing for Keyboard Accessibility](#)

[3. Interactive HTML Elements and Key Presses](#)

[Native HTML Elements & Common Key Presses](#)

[Customized Elements & Uncommon Key Presses](#)

[Custom Interaction Example: Balloons and Static Electricity](#)

[4. What Does Not Receive Keyboard Focus](#)

[5. Parallel DOM Structure and its Role in Keyboard Access](#)

[Example PDOM for Balloons and Static Electricity \(BASE\)](#)

[Looking at labels alone for focusable Items](#)

[Example Template for PDOM Headings](#)

[6. When to Begin Designing Keyboard Access](#)

[7. Grouping things to Create Relationships \(added April 9, 2017\)](#)

[Technical Notes on Groupings](#)

[Radiogroup Role](#)

[Menu, Menubar and Toolbar Roles](#)

[Questions still to answer about groupings using ARIA techniques:](#)

[8. Resources](#)

[Practice](#)

[Capacitor Lab: Basics](#)

[Proportion Playground](#)

1. Keyboard Accessibility: An Introduction

Keyboard accessibility is concerned with:

- Navigating interactive sim objects in the sim and,
- Navigating and reading through in-sim or in-page content.

For visual keyboard users (i.e., a user that does not use a mouse, and also does not use a screen reader), our main concern is navigating the interactive sim objects. A user must be able to navigate to, and activate, all of the interactive sim objects using keyboard presses alone, and

then operate the interactive sim object with key presses that are either familiar to them or provided to them.

2. Process for Designing for Keyboard Accessibility

- 1) List the sim’s interactive objects - all objects requiring the ability to have keyboard focus.
- 2) For each object, determine if the interaction with the object:
 - a) maps easily to native interactive HTML elements (links, form elements, controls).
 - i) For example, PhET Sims avail of many sliders and switches which have native HTML elements and ARIA roles.
 - b) maps easily to an existing custom interaction used in a keyboard accessible PhET sim
 - c) does not map easily to a native interactive HTML element, nor an existing (PhET) custom element - requiring a new custom interaction or design change.
- 3) Determine ~~order of headings within PDOM, and~~ order of focusable sim objects.
- 4) Determine any visual keyboard states (e.g., Hover, Focus, and Active states) that need to be used to communicate what is happening during keyboard interaction.
 - a) New visuals may be needed to support custom interactions.
- 5) Include all relevant key presses and custom interactions in the sim’s keyboard help dialog.

3. Interactive HTML Elements and Key Presses

All interactive sim objects will receive keyboard focus and need an equivalent HTML element in the Parallel DOM. Links, form elements, and controls, are interactive HTML elements, natively. These common interactive elements are operable with standard key presses that are well-known to keyboard users and screen reader users. Mouse users may not be familiar with native key presses.

Mapping interactive sim objects to a native HTML element is the most straightforward way to make the interactive sim object focusable and intuitively operable with common key presses.

Native HTML Elements & Common Key Presses

Interaction	Keystrokes	Notes
Navigate focusable elements	<ul style="list-style-type: none"> • Tab • Shift + Tab - navigate backward 	<ul style="list-style-type: none"> • Visual keyboard focus indicators must be present. • Navigation order should be logical, intuitive, and pedagogically relevant.

Link	Enter	
Button	Enter or Spacebar	<p>Ensure elements with ARIA role="button" can be activated with both key commands.</p> <p>An explicit ARIA button role is not needed needed on a native button. It has the button role is there by default.</p>
Checkbox	Spacebar - check/uncheck a checkbox	Checkboxes should be used when one or more option can be selected.
Radio group with radio buttons	<ul style="list-style-type: none"> • ↑/↓ or ←/→ - select an option. • Tab - move to the next element. 	<p>Radiogroup should be used when only one option from a group can be selected.</p> <p>Radio groups are likely best when there are more than two choices. Consider a switch role when there are only two choices.</p> <p>Radio groups have a key press pattern that may seem unintuitive to people who can see. See ARIA Authoring Practices 1.1 section 2.16</p>
Select box (form control with options that dropdown)	<ul style="list-style-type: none"> • ↑/↓ - navigate between options • Spacebar - expand 	You can also filter by typing letters, but this behavior varies by browser. Some will filter as you type, like autocomplete. Others will only sort by first letter.
Dialog	Esc - close And a focusable visual close button.	<ul style="list-style-type: none"> • Modal dialogs should maintain keyboard focus. • Non-modal dialogs should close automatically when they lose focus. • When a dialog closes, focus should usually return to the element that opened the dialog.
Slider	<ul style="list-style-type: none"> • ↑/↓ or ←/→ - increase or decrease slider value 	<ul style="list-style-type: none"> • For double-sliders (to set a range), Tab/Shift + Tab should toggle between each end. • In some sliders PageUp/PageDown can move by a larger increment (e.g., by 10).

	<ul style="list-style-type: none"> • Home/End - beginning or end 	
Spin Box	<ul style="list-style-type: none"> • ↑/↓ or ←/→ or Home/End • number entry 	<ul style="list-style-type: none"> • Spinbox includes editable value, so the user could type in a value directly.

*Table adapted from [WebAIM](#)

*Note the most up-to-date technical reference for standard key presses is the [ARIA 1.1 Authoring Practices](#).

Customized Elements & Uncommon Key Presses

In cases where an interactive sim object cannot be mapped to a native HTML element, a customized approach is necessary.

Custom Interaction Example: *Balloons and Static Electricity*

The balloon in the *Balloons and Static Electricity* sim. There is no native interactive HTML element that uses the arrow keys to move an object in four directions.

- **Solution:** We used a native control element, a button, to activate (or grab) the balloon. We then implemented a custom interaction where the arrow keys move the balloon in four directions, something that is intuitive, but not a common or “standard” interaction. In this case, we additionally implemented custom key presses (the W, A, S, and D keys) to provide operability for some screen readers. A combination of a native HTML element with customized key presses allowed relatively intuitive access to and interaction with the balloon.

If an interaction cannot be mapped to a native HTML element, and cannot be mapped to an existing PhET custom interaction, let Emily, Taliesin, and Jesse know.

[Future Update - List of PhET's Custom Interactions]

Note that we started a document about 2 years back when we just learning about ARIA. We should dig that document up to see if we can make more use of it now, especially since we know a lot more.

4. What Does Not Receive Keyboard Focus

There is a difference between “keyboard focus” and “keyboard access”. For example, sims can have dynamic displays that provide information to the user, but are not directly operated by the user. The Sweater and the Wall in the *Balloons and Static Electricity* sim are examples of dynamic displays. The Sweater and the Wall dynamically display information to the user, but the user interacts with these displays indirectly via the balloon. Dynamic display information would need to be described to a student who cannot see, but not to a student who can. Thus, sim

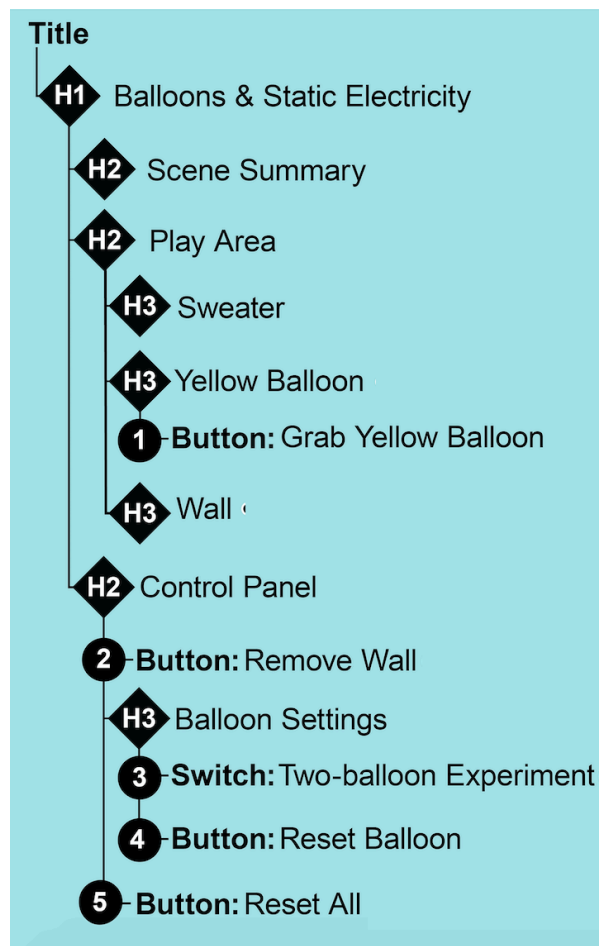
objects that only display information and are not directly interacted with do not receive keyboard focus.

In the design process, we may want to consider how the displays might be described to the non-visual user, but descriptions of dynamic changes is not needed for visual keyboard access.

5. Parallel DOM Structure and its Role in Keyboard Access

Example PDOM for Balloons and Static Electricity (BASE)

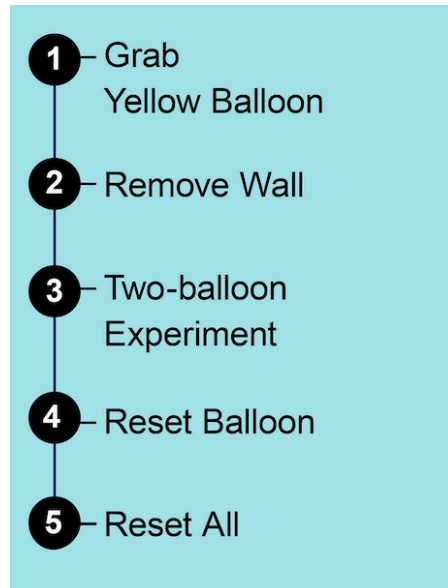
Here is an example of the high level structural representation of the PDOM for BASE (note without the Radio Group for the three charge views). Together the headings and the labels form an outline of all the objects in the sim, including three main regions or areas (Scene Summary, Play Area, Control Panel) that are new. The headings and many labels are not shown visually in sim. They do, however, form a very useful starting point for the design of description for non-visual access. Some headings (e.g., H3: Sweater and H3: Wall) act as labels for these two objects that dynamically display information, but are not directly interacted with.



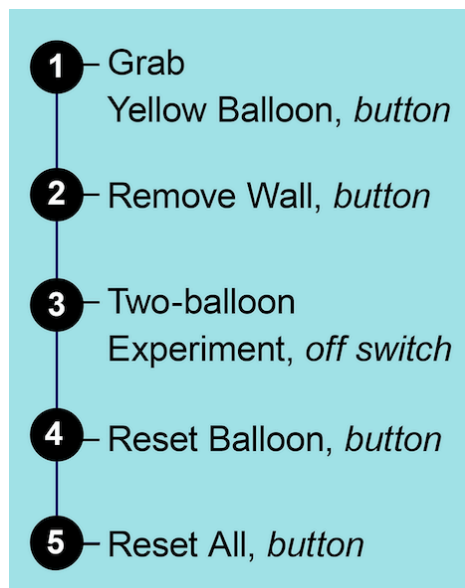
Headings and labels do not change, so it is useful to consider what they will be early in the design process.

Looking at labels alone for focusable Items

Every interactive sim object will be represented by an interactive HTML element in the PDOM, and will require a label, even if the label is not read out to visual keyboard users.







Here are the same labels as they would sound with structural information that is provided when using a screen reader:







In some cases, labels (and even the choice of the interactive HTML control) may need to be adjusted to sound better when read together with structural information. Item 3, is a good



example. The label and structure changed in iterative designs for BASE to make it more concise while at the same time more contextual and playful within the outline of the sim.

Remove/Add Green Balloon radio buttons	
	
Radio button checked, <i>Remove green balloon, 1 of 2</i>	Radio button not checked, <i>Remove green balloon, 1 of 2</i>
Radio button not checked, <i>Add green balloon, 2 of 2</i>	Radio button checked, <i>Add green balloon, 2 of 2</i>

Add/Remove Green Balloon toggle button	
	
<i>Add green balloon,</i> toggle button, not pressed	<i>Remove green balloon,</i> toggle button, pressed

Two-balloon Experiment toggle button	
	
<i>Two-balloon experiment,</i> toggle button, not pressed	<i>Two-balloon experiment,</i> toggle button, pressed

Two-balloon Experiment with switch role	
	
<i>Two-balloon experiment, off</i> <i>switch</i>	<i>Two-balloon experiment, on</i> <i>switch</i>

Two-balloon Experiment with switch role	
	
<i>Two-balloon experiment, off</i> <i>switch</i>	<i>Two-balloon experiment, on</i> <i>switch</i>

Example Template for PDOM Headings

- H1: **Sim Title**
- H2: **Scene Summary**
- H2: **Play Area**

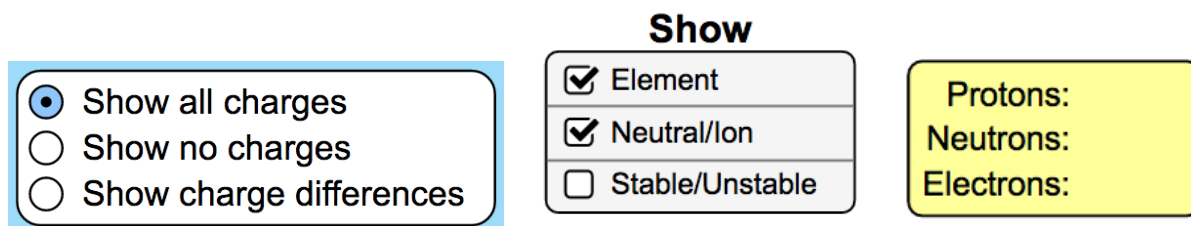
- *The Play Area is where the students interact with interactive sim objects.*
- Designer to consider and possibly identify first draft sub-headings for dynamic display objects.
- H2: **Control Panel** (can think of this as “Control Area or Areas”)
 - *The Control Panel is where students use controls to add and remove things from the Play Area, and/or set parameters for the interactive objects.*
 - *Subpanels may be needed to group controls within the Control Panel for efficient navigation.*
 - Designer to identify first draft labels for interactive controls.

6. When to Begin Designing Keyboard Access

- Once the layout of the sim screen(s) and all major interactions are set, the following can be drafted. Drafts can be refined as sim design is updated.
 - Identification of interactive objects
 - Classification as standard, non-standard but matches existing PhET customization, or needing new custom interaction
 - Draft of PDOM structure, and identifying potentially non-standard interactions
 - For non-standard interactions, the potential need for visual cues should be considered
- As sim nears completion:
 - Finalize classifications, PDOM structure, and any new non-standard interactions
 - Finalize any visual cues
 - Update (if needed) and finalize keyboard navigation dialog

7. Grouping things to Create Relationships (added April 9, 2017)

Questions about grouping have come up. Often in PhET Sims, related content and controls such as radiobuttons, checkboxes and/or other items are visually grouped. Here three simple examples from *Balloons and Static Electricity* and *Build an Atom* simulations:



Other more complicated or nuanced groupings can also be found in PhET Sims.

COULD provide more examples here, if needed.

- There may be some useful discussion on this issue in the github issue [balloons-and-static-electricity/issues#213](https://github.com/phetsim/balloons-and-static-electricity/issues/213) where we discussed scene selection buttons.

Groupings help create context visually and non-visually, and depending on the design can affect how content is delivered and accessed. In the design process it may be useful, even important to consider the following:

- Are items in the group individual interaction points?
- How important/critical is the interaction point to the learning goals of the sim?
- Is the the group itself an interaction point? If so, what is the level of importance of the group?
- Should a name for each item in the group be available in the outline of the simulation?
- Should a common name for the group be available in the outline of the simulation?
- Should some combination of the above be used?

HTML and ARIA both offer several grouping and labeling techniques to help communicate relationships in the code. Unfortunately, not all grouping techniques are supported by browsers and assistive technology in the same way or even at all. Some solutions are more wordy or verbose than other solutions when it comes to how AT read out the information. Panchang (2012) provides [a comparison](#) between using the HTML fieldset and legend elements and using the ARIA group role with an aria-label. In this example, group and label information of both techniques are communicated almost the same way by screen reader software. The point of the example is to show that the group role is safe to use. It may be quite using in building out interactive widgets for PhET sims.

The 2017 article [What is an accessible name?](#) By Léonie Watson may be helpful, so just adding it here for now.

Technical Notes on Groupings

This document should focus on design considerations rather than technical issues, but the 2, are kind of related when it comes to accessibility.

Radiogroup Role

I thought radiogroup was both an HTML element and an ARIA role, but I could not find the element in the HTML5 specification. I only found the following examples on MDN

- [input types](#)
- [radiogroup](#)

I have a feeling that only the ARIA role “radiogroup” is supported in browsers.

Menu, Menubar and Toolbar Roles

- May need to bring in thoughts and resources from Git Hub issue [a11y-research #25](#)

The ARIA roles menu, menubar, and toolbar are other roles that can be used to group items and to change the interaction pattern; however, at this time we need to use these roles with caution as their support in browsers and AT still vary quite a lot. These roles have quite a specialized function. They are meant to make things work like menus and toolbars in desktop applications, so we need be sure that's what the design requires and that it works. We design and build a custom interaction when the standards-based pattern either is not supported in browsers, does not fit the design, or both.

Questions still to answer about groupings using ARIA techniques:

1. Do ARIA-labels on non-interactive elements get added to the sim outline? If not, in some case we would want to use headings and *aria-labeledby* instead.
2. In what situations does a grouping warrant a change in keyboard interaction (i.e., use of Arrow keys to navigate the items). For example, I have found no examples where this is true for checkboxes.

8. Resources

- WebAIM [Introduction to Keyboard Accessibility](#)
- Smith, Taliesin. (2014) [RAMP It UP! Action-base guide to building accessible websites](#)
- Faulkner, Steve. (2016) [Notes on ZoomText Web Finder](#)
- Smith, T., Lewis, C., and E.B. Moore. (2016) [Inclusive Interaction: Description strategies for a complex interactive science simulation](#)
- Watson, Léonie. (2016) [What does accessibility mean?](#) Paciello Blog.
- [HTML5 A11y Support](#)
- [Notes on using ARIA](#) (2016)
- Panchang, Sailesh. May 2012, [ARIA-Group: a viable alternative for Fieldset / Legend?](#) Deque Blog.
- Note see email April 10th with some technical questions.

Practice

[Capacitor Lab: Basics](#)

Interactive Objects (Screen 1):

2 Circuit Switches (Radio group??)

Battery voltage slider (Native HTML Slider)

4 Check boxes in control panel (Native HTML Checkbox)

Plate Charges

Bar graph

Electric Field

Current Direction

Plate separation slider (Native HTML Slider)

Plate area slider (Native HTML Slider)

Voltmeter

Body

2 probes

Reset All Button

Classification of Objects:

PDOM Structure:

[Proportion Playground](#)

Interactive Objects:

Classification of Objects:

PDOM Structure: