Keyboard and Screen Reader Accessibility in Complex Interactive Science Simulations: Design Challenges and Elegant Solutions

Emily B. Moore(✉), Taliesin L. Smith, and Jesse Greenberg

University of Colorado Boulder, Boulder, USA
{emily.moore,taliesin.smith,jesse.greenberg}@colorado.edu

Abstract. Interactive science simulations are commonly used educational tools that, unfortunately, present many challenges for robust accessibility. The PhET Interactive Simulations project creates a suite of widely used HTML5 interactive science simulations and has been working to advance the accessibility of these simulations for users of alternative input devices (including keyboards) and screen reader software. To provide a highly interactive experience for students, science simulations are often designed to encourage interaction with real-world or otherwise physical objects, resulting in user interface elements being implemented in ways either unrecognizable as native HTML elements, or that require fully custom implementation and interactions. Here, we highlight three examples of simulation design scenarios that presented challenges for keyboard and screen reader access. For each scenario, we describe our initial approach, challenges encountered, and what we have found to be the most elegant solution to address these challenges to date. By sharing our approaches to design and implementation, we aim to contribute to the general knowledge base of effective strategies to support the advancement of accessibility for all educational interactives.

Keywords: Web accessibility · Usability · Inclusive design
Keyboard navigation · Alternative input · Text description
Interactive science simulation

1 Introduction

Interactive simulations are effective tools used to support teaching and learning of science content around the world (e.g., [1]). Unfortunately, the vast majority of interactive simulations are not accessible to many students. A reliance on highly visual representations and the use of mouse, trackpad, or touch interfaces limits the accessibility of these learning resources for students with certain sensory, mobility, or cognitive impairments. While there has been progress in the development of standards and guidelines to support the accessibility of interactive scientific graphics [2], and digital games [3], many of these resources do not address the specific design and implementation challenges encountered within the context of creating accessible complex interactives for learning [4, 5].

The PhET Interactive Simulations project [6] at the University of Colorado Boulder, a resource that includes more than 50 popular free HTML5 science and mathematics simulations, has been designing and implementing multiple new accessibility features into simulations to support access for students with disabilities [7]. These accessibility features include auditory description (verbalized text), sonification (non-speech sound), and alternative input. PhET simulations are complex from an accessibility perspective for multiple reasons and require innovative approaches to provide the most inclusive outcome. Here, we focus on three design challenges that arise when creating accessible complex interactive science simulations, and some of the more elegant solutions we have found to address these challenges. All solutions can simultaneously support visual and non-visual access, consistent with a goal of inclusion and the practical need to support collaborative learning opportunities for students with diverse needs.

2 PhET Interactive Simulations

PhET simulations (sims) are designed to be playful, intuitive to use without instructions, and flexible for use in many different teaching contexts (lecture, group activities, online, etc.). The sims are designed to provide strong visual cues to support effective use. Interactive objects, layout, and colors are chosen to create an inviting environment for students of all ages. The sims do not contain explicit instructions, which allows for tremendous flexibility of use – each sim can be used to meet a range of learning goals across different grade levels and topics. Each sim's design supports initial exploration, which results in pedagogically relevant changes and encourages further investigation.

As an example, we introduce the sim *Balloons and Static Electricity* (Fig. 1). This sim is used from elementary school through introductory college level to introduce the topic of static electricity, charge transfer, attraction, repulsion, and induced charge. The

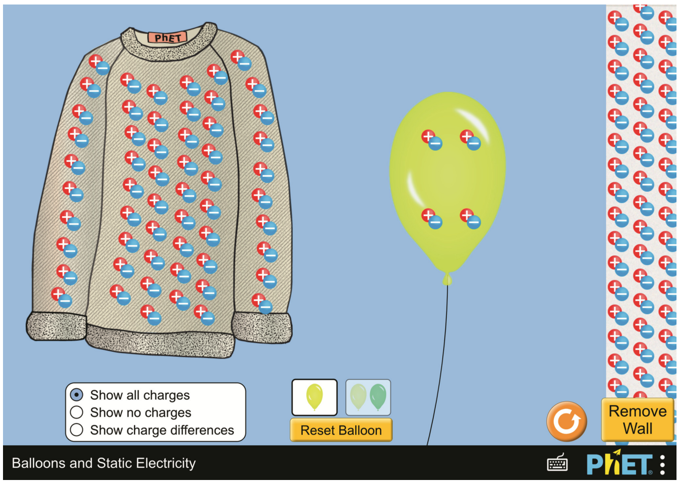

Fig. 1. Screenshot of the PhET simulation *Balloons and Static Electricity*. Reproduced with permission from the PhET Interactive Simulations project (Color figure online).

sim opens with a Yellow Balloon in the center of the screen, a Sweater to the left and a Wall on the right. On each object are pairs of positive charges (red circles with "+" on them) and negative charges (blue circles with "−" on them). Near the bottom of the screen, there are options to change the view (to show all charges, show no charges, or show only the charge differences), to add/remove a second (green) Balloon, to reset the Balloon to its starting location, to remove/return the Wall, and to reset the entire sim screen.

The Balloon can be moved around the sim screen in all directions, including rubbing on the Sweater (resulting in a transfer of negative charges from the Sweater to the Balloon), and rubbing against the Wall (resulting in no transfer of charges). When the Balloon is negatively charged, releasing the Balloon will result in it being attracted to the Sweater or the Wall, depending on the amount of charge and its proximity to these other objects – the more negatively charged the Balloon is and the closer to an object, the faster it moves.

The *Balloons and Static Electricity* sim is one example of many PhET sims available. Each PhET sim is unique and tailored to encourage productive exploration of a specific science or mathematics topic. The result is a suite of simulations that support learning [8, 9], utilizing many different interactive objects (balloons, sweaters, and walls, as well as arms, planets, wire, atoms and more), through many different interactions.

2.1 PhET Simulation Architecture

Each sim's graphical representations are organized and rendered using the custom scene graph, Scenery. With Scenery, rendering of a PhET sim is done graphically with multiple renderers, including SVG, Canvas, and WebGL. Use of a scene graph provides a logical data structure, but does not contain – or provide a straightforward way to associate – the semantic structure typical of HTML. Graphics created with a single renderer (e.g., SVG), such as chemical diagrams and data charts, have been made accessible through the addition of descriptions and simple hierarchical navigation [10, 11]. However, in the case of PhET sims, where multiple graphics renderers are used and frequent interaction is paramount, the challenge of adding accessible descriptions and interactivity is more complex. To address this challenge, we advanced Scenery to now generate a parallel document object model (PDOM) alongside the graphical representations [12]. The result is a robust accessible – and dynamic – document which contains the semantically rich interactive content (e.g., regions, headings, labels, help text, object descriptions, and alerts) and interfaces with common assistive devices with appropriate use of HTML and WAI-ARIA [13–15]. The PDOM solution avoids current shortcomings of accessibility solutions specific to each graphics renderer and provides a wide range of native interactions. Significant use of native interactions allows us to limit the use of custom approaches that require the use of the ARIA application role which can result in unintuitive interactions – and present a potential barrier to access – for users.

2.2 Accessibility Challenge

The inclusive design of navigation and interaction capabilities for alternative input (such as keyboard input) for highly interactive and highly dynamic sims involves addressing visual and non-visual accessibility needs simultaneously. Alternative input may be used to directly interface with a sim or may be used to control software – such as screen reader software, that is providing access. Screen reader software is used to navigate applications, interact with interactive elements, and read aloud text content – including onscreen text presented visually, as well as content provided non-visually, e.g., alternative text, HTML element labels, and help text.

Complex interactive simulations present many challenges for alternative input capabilities and for screen reader compatibility. Elegant solutions that simultaneously support visual and non-visual access within complex interactives can be challenging to design and implement. In this work we present three design challenges and the solutions we arrived at. These solutions leverage native HTML enhanced with WAI-ARIA to design and implement complex accessibility features for interactive graphics objects.

2.3 PhET's Design Process

The PhET project conducts all design and development work through an iterative process involving user interviews (e.g., [5, 16]). In the development of the accessibility features presented in this work, we refined our approaches through user interviews with visually impaired users, and consultation with expert screen reader users. User interviews are conducted in a think-aloud style, where an accessible sim is provided to a user not familiar with a particular sim and they are prompted to think-aloud while using the sim. The interviewer may ask clarifying questions as the user engages with the sim, but the primary goal of the interviewer is to observe and listen. Once the user is done exploring the sim – the bulk of the interview time – the interviewer will ask further clarifying questions and may ask the user to attempt specific tasks with the sim to gain further insight into the design of particularly challenging interactions or representations. When consulting with screen reader experts, we gain insight from users knowledgeable about PhET sims and typical interactions, which provides an opportunity to ensure that the navigation and description choices we make are perceived as consistent across sims and screen reader software.

3 Design Challenges and Solutions

This work presents design challenges we have encountered and our implemented solutions – focusing on scenarios that highlight design to support visual and non-visual access. For each challenge we summarize the specific goals of a desired interaction sequence, the challenges we sought to address, and our implemented solution. All challenges and solutions take into consideration interviews with users with and without visual impairments, discussions with expert screen reader users, as well as the general expertise of PhET's design team and pedagogical expertise of PhET's science content and pedagogy experts. The result is that some design decisions and aspects of the resulting solutions stem from unique pedagogical needs of the sims and may not be a direct result from interpreted or stated user needs or preferences from interviews.

3.1 Custom Slider Features to Support Content Learning

The sim *Resistance in a Wire* (Fig. 2) is used in high school and introductory college physics courses to teach the relationship between the resistance of a wire and its resistivity, length, and area. The sim consists of an equation ($R = \rho L/A$) and a piece of a wire, with three sliders with various input ranges to adjust the values of resistivity (ρ), length (L), and area (A). Incrementing or decrementing the slider range values results in a change to the size of the variables in the equation and parameters of the piece of wire. For example, increasing the length slider value results in an increase in size of the variable *L* in the equation and a corresponding increase in size of the variable *R*, while simultaneously increasing the length of the piece of wire. Decrementing the resistivity slider value results in a decrease in the size of the variables ρ and *R* in the equation, and decreasing the number of small black dots in the piece of wire, which represent impurities.

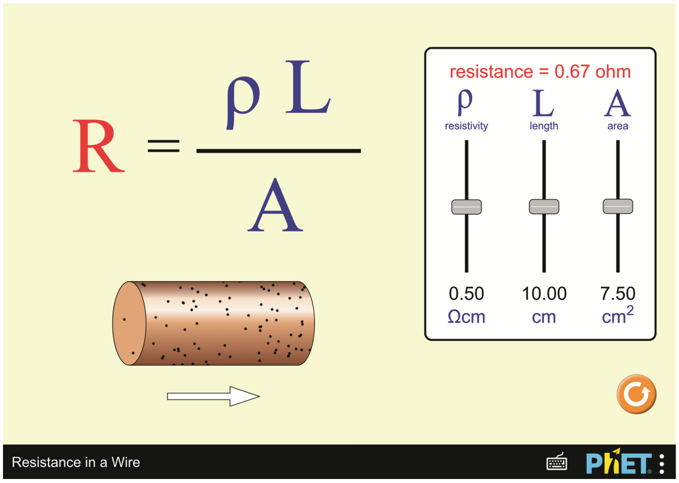

Fig. 2. Screenshot of the PhET sim *Resistance in a Wire*. Reproduced with permission from the PhET Interactive Simulations project.

The desired interaction sequence with this sim is for students to initially explore by changing the sliders' input values with a qualitative focus, changing the input values to extremes (e.g., the maximum, minimum, and median values) and observing how the variables and the piece of wire change. Then, as students continue manipulating the sliders, to compare changes across the mathematical and physical representations, building connections between the mathematical equation and a real-world example (the physical piece of wire). Another useful interaction sequence would be for students to make quantitative comparisons, using the numerical slider values to set up comparisons convenient for simple mental calculations – typically doubling or halving values. For example, comparing two scenarios with the same resistivity, but one scenario having double the length and area as the other scenario (both scenarios result in the same resistance). Attending to the quantitative slider values can also be utilized by teachers in the classroom to direct students to specific scenarios for exploration, calculations, or discussions.

When implementing alternative input capabilities and descriptions, we wanted to ensure we provided access to the sliders, and that this included intuitive and efficient access to: (1) the extreme values of each input range, with description highlighting the extreme size differences in the equation variables that result, (2) "double" and "halve" slider values, and (3) all values in the input range.

Initial Implementation. All interactive objects in the sim were instrumented to support alternative input access and descriptions were designed and implemented to support non-visual access. Here we focus on the navigation and description capabilities associated with the sliders.

We initially implemented each slider with the standard and recommended native HTML slider attributes to support access to the input type range. Users could press *Tab,* or other common navigation approaches, to move focus to each of the three sliders. Once a slider had focus, Arrow keys could be used to increment (*Up/Right Arrow keys*), or decrement (*Down/Left Arrow keys*) the input values. The initial step size for each slider was 1/100th of the slider range. Using the *Home/End* keys resulted in jumping to the maximum/minimum input range. Using the *Pg Up/Pg Down* keys resulted in incrementing/decrementing the input value by 1/10th of the slider range. As the slider is adjusted by the student, a description of the resulting slider input value is provided, followed by an alert describing the relative size of the equation variables – e.g., "As letter ρ grows, letter R grows. Resistance is now 0.82 Ohms."

Challenges. The initial implementation effectively supported efficient access to the maximum and minimum slider values (through the use of *Home/End*, or alternatively, multiple presses of the *Pg Up/Pg Down* keys), but did not provide efficient access to easily doubled or halved input values, or to all possible input values. These issues are a result of the extreme differences in the input range for the ρ (resistivity) slider as compared to the length and area sliders (see Table 1). To support effective qualitative comparisons, the range must span values that students perceive as physically small to large (e.g., length ranging from 0.10 cm to 20.00 cm). Minimum values must be non-zero to avoid zero or undefined resistance.

Table 1. Initial input range parameters for sliders in *Resistance in a Wire*

	Resistance	Length	Area
Input range	[0.01, 1.00]	[0.10, 20.00]	[0.01, 15.00]
Median value	0.5	10.00	7.5
1/10 range	0.099	1.99	1.49
Min step size	0.01	0.01	0.01
Total number of input values	99	1990	1499

In the initial implementation, using the arrow keys to adjust the input values would result in default step sizes of 0.0099 for ρ, 0.199 for Length, and 0.149 for Area, as typical HTML range inputs require the range to be evenly divisible by the step size for the whole range of values to be accessible. These are not intuitive numbers for comparisons, or mental mathematical manipulation like double or halving. Figure 3 compares the slider panel in *Resistance in a Wire* with quantitative values not rounded (Fig. 3, left panel), and rounded (Fig. 3, right panel). Such a small default step size could also have the undesirable outcome of initial changes to the input value resulting in very small changes to the equation and piece of wire, obscuring important qualitative comparisons readily apparent with larger changes to the input values.

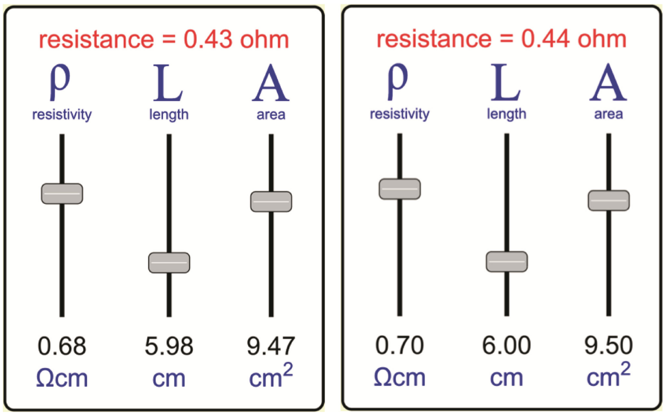

Fig. 3. Slider panel from the PhET simulation *Resistance in a Wire* showing (left panel) quantitative values that are not rounded and (right panel) quantitative values that are rounded.

Solution. To address these challenges, we first implemented the sliders with the standard native HTML features, and then added two additional new features: a modifier key and a custom variable step size. The new modifier key is the *Shift* key. Used in combination with the Arrow keys, i.e., *Shift* + Arrow key, increments/decrements the input value by the smallest allowable step size. For this sim, across all sliders, this is 0.01. Using standard ways of adjusting the input value of a slider – Arrow keys, *Home/End*, *Pg Up/Pg Down* – allow the user to efficiently span the range of input values, and use of the *Shift* + Arrow keys allows fine control when targeting a specific input value. We will implement this modifier key for all sliders in PhET sims, for consistency.

A variable step size was also implemented, to support more efficient access to input values that support simple mental calculations. There is a base step size used through most of the input range. This base step size is varied to address two specific scenarios – ensuring the minimum and maximum values of the sliders can always be reached, and ensuring that rounded input values (easier for mental mathematical calculations) are reached when transitioning between different slider navigation approaches. For the ρ slider, with a range of 0.01–1.00, the base step size is used across the range 0.05–1.00. When decrementing with an Arrow key from an input value of 0.05, the input value changes to 0.01 the minimum input value (a step size of 0.04) – ensuring the Arrow keys support navigation to the extreme ends of the input range. When using the *Pg Up*/*Pg Down* keys, or the *Shift* modifier key to access an input value not an increment of 0.05, incrementing/decrementing with an Arrow key will result in an input value that is the nearest increment of 0.05. Similarly, the Length and Area sliders have a base step size of 1.00. When decrementing by Arrow key from an input value of 1.00, the input value changes to the minimum value – 0.10 (Length) and 0.01 (Area). When an input value is accessed that is not an increment of 1.00, incrementing/decrementing by Arrow keys results in the nearest input value that is an increment of 1.00.

The result of this implementation are sliders that can be accessed in all of the typical ways – meeting expectations of interaction with sliders for users with and without visual impairments – and that address the pedagogical goals of supporting access to (1) the extreme values of each input range, (2) values easy to "double" and "halve", and also (3) all values in the input range.

3.2 Interacting with Uncommon Objects Using Native HTML Elements

The sim *John Travoltage* (Fig. 4) is used from elementary school through introductory college level to introduce the topic of static electricity. The sim opens with the character John, standing in a room, on a rug, next to a door. He has a moveable leg and arm. Rubbing his foot on the rug results in electrons, represented as blue circles with a "–" sign on them, transferring from the rug onto John's body. John's arm can be moved closer, or farther, from the nearby doorknob. Moving John's arm closer to the doorknob can result in the electrons on John's body discharging into the doorknob, resulting in a shock. A productive exploration of this sim typically includes students exploring the relationship between the amount of charge on John's body, and the distance between

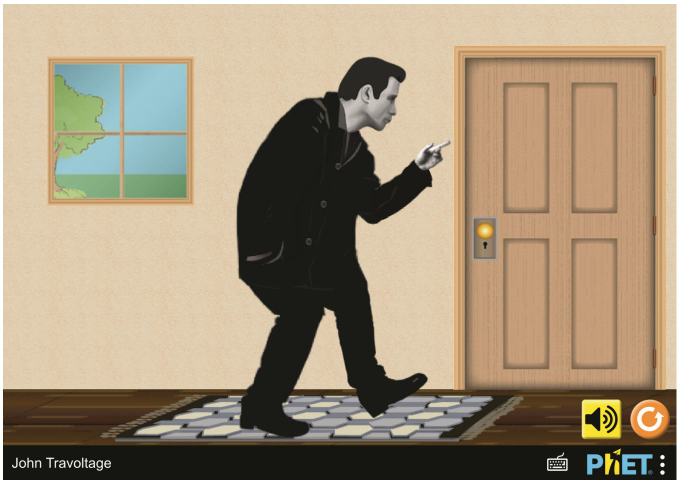

Fig. 4. Screenshot of PhET sim *John Travoltage*. Reproduced with permission from the PhET Interactive Simulations project.

his hand and the doorknob that result in a discharge. The more charge on John's body the farther his hand can be from the doorknob while still getting shocked.

When implementing alternative input capabilities and descriptions, we wanted to ensure (1) leg and arm interaction is intuitive – since these are the primary interactive objects in the sim, (2) efficient access to electron transfer and discharge scenarios, and (3) descriptions support users in knowing the relative location of the arm/leg.

Initial Implementation. In collaboration with the Inclusive Design Research Centre at OCAD University, all interactive objects in the sim were instrumented to support alternative input access and to provide description. Here we focus on the navigation and description capabilities associated with the arm and leg. When the sim was first developed, the arm and leg objects were not implemented as HTML elements. The arm and leg each spanned a range of locations – 180° of a circle for the leg, and 360° of circle for the arm. We investigated implementing the arm and leg as sliders, number spinners, and in an application mode. Upon consideration, we concluded sliders would likely provide the most intuitive experience to the desired interaction pattern [17].

Our initial implementation of these objects as sliders had the arm implemented with a range of 0–60, and the leg with a range of 0–30, each with a step size of 1. Descriptions indicated 12 different regions of the slider (Fig. 5, left panel), with each region spanning

Initial Implementation

Position Number	Region Descriptions
0	Farthest from the doorknob.
1 - 5	Very far from the doorknob.
6 - 10	Far from the doorknob.
11 - 15	Neither far or close to the doorknob.
16 - 20	Somewhat close to the doorknob.
21 - 25	Close to the doorknob.
26 - 29	Very close to the doorknob.
30	Closest to the doorknob.
31 - 34	Very close to the doorknob.
35 - 39	Close to the doorknob.
40 - 44	Somewhat close to the doorknob.
45 - 49	Neither far or close to the doorknob.
50 - 54	Far from the doorknob.
55 - 59	Very far from the doorknob.
60	Farthest from the doorknob.

Current Implementation

Position Number	**Landmark** & Region Descriptions [RD]
-15	**Farthest from the doorknob. Last stop.**
-14	Very far from the doorknob.
-13	**Hand pointing away from door**, [RD].
-12	Very far from the doorknob.
-11	Far from the doorknob.
-10	
-9	
-8	**Hand pointing straight up.**
-7	Not so close to doorknob.
-6	Not so close to doorknob.
-5	Close to doorknob.
-4	**Hand pointing at upper door**, [RD].
-3	Close to doorknob.
-2	Very close to doorknob.
-1	Just above doorknob.
0	**At doorknob.**
1	Just below doorknob.
2	Very close to doorknob.
3	Close to doorknob.
4	**Hand pointing at lower door**, [RD].
5	Close to doorknob.
6	Not so close to doorknob.
7	**Hand pointing straight down.**
8	Not so close to doorknob.
9	Far from doorknob.
10	
11	
12	Very far from doorknob.
13	**Hand pointing away from door,** [RD].
14	Very far from doorknob.
15	**Farthest from the doorknob. Last stop.**

Fig. 5. Position, region, and landmark descriptions for the initial implementation (left) and current implementation (right) of the arm slider in the PhET simulation *John Travoltage*.

five input values. Eleven of these regions had one description that would be read aloud, indicating proximity to the doorknob, e.g., “Position 12, Close to the doorknob” and “Position 55, Far from the doorknob.” In the region nearest the doorknob, additional descriptions were provided, indicating “Close to doorknob,” “Very close to doorknob,” and “Closest to doorknob.” Descriptions for the leg, which moves 180° of a circle, had three regions – each indicating whether foot was off (e.g., “Position 2, Foot is off the rug”), or on the rug (e.g., “Position 15, Foot is on the rug”).

Challenges. The slider implementation successfully supported access to incrementing and decrementing arm and leg position values. The interaction did not support efficient access to charge transfer or discharge behavior, and the descriptions were not always meaningful during interaction. A common complaint from users was that it took too many keypresses to reach a new description region, resulting in repetitive descriptions and an overall feeling that the arm/leg was moving too slowly. Also, each object has a ‘target’ area, a location that has unique consequences for the slider interaction. For the arm, this is typically the region where the hand is nearest the doorknob; for the leg this is the region where the foot is on the rug.

Additionally, there are two challenges inherent in using a linear slider to represent a circular interaction. First, there is no way to continuously increment the input values to result in multiple same-direction (e.g., clockwise) passes around the circle. For example, incrementing the slider from input value 0 to 60 results in moving the arm clockwise around the circle. Once the maximum input value of 60 is reached, the user must stop, or decrement through decreasing input values (moving arm counterclockwise) to continue moving the arm. The second challenge is that at some point in traversing the slider input values, using the *Up/Right* (or *Down/Left) Arrow* key to increment (or decrement) the input value will actually result in decrementing (or incrementing) the position value.

Solution. To address these challenges, we decreased the input value ranges for the arm and leg, and made significant changes to the description approach. For the arm, the input value range is now -15 to 15; the leg has an input range of half that, -7 to 7. This results in requiring fewer keypresses to span the full range, and more object movement with each keypress.

To improve the descriptions of the arm position, three description types were created (Fig. 5, right panel) the region (e.g., “Close to doorknob,” “Not so close to doorknob,” “Far from doorknob”), landmark (e.g., “Hand pointing straight up” and “At doorknob”), and direction change descriptions (i.e., “Away from doorknob” or “Towards doorknob”). Upon change of position for the arm, the slider value is announced as well as one of the following: If change of direction, direction is announced; if arm enters landmark location, landmark is announced; otherwise, region is announced. The leg descriptions were also improved. For the leg, one of two regions are announced in addition to slider value: “Foot rubbing on rug” or “Foot off rug”.

The resulting sim interaction provides efficient access to the leg and arm as well as the important electron discharge scenarios, while the descriptions provide new information on relative position with each move of the leg or arm. The result is an interaction that highlights the significance of the location of the arm with respect to the doorknob, supporting the user to create and explore electron discharge events.

3.3 Providing Custom Control and Using Visual and Non-visual Cues

The sim *Balloons and Static Electricity* (described in Sect. 2, shown in Fig. 1) allows students to investigate static electricity, charge transfer, attraction, repulsion, and induced charge through free range movement of a Balloon. The Balloon can be rubbed on a Sweater or a Wall, resulting in different charge transfer or induced charge behavior. The Balloon can be moved by the user, and can also move independently upon release – with the Balloon location, state (more or less negatively charged), and release location all being significant for the free-movement velocity of the Balloon, and the final resting position of the Balloon.

When implementing alternative input capabilities and descriptions, we wanted to ensure students could (1) intuitively pick up, move, and release the Balloon, (2) easily understand the specific location of the Balloon, and (3) understand when a transfer of negative charge occurs.

Initial Implementations. The *Balloons and Static Electricity* sim presented numerous challenges for implementing accessibility features. The Balloon in the sim needs to have a four-way, drag and drop-like, behavior – with any location in the sim being a possible release location. Upon release the Balloon may move to other locations based on the current state of charge differences in the sim. This behavior is unlike any native HTML elements [13]. Additionally, while the Balloon is in motion (either controlled by the user or moving independently) the user needs to be kept up-to-date on various state changes, including the Balloon's current location, any occurrence of charge transfer and the resulting net charge, any occurrence of an induced charge at the wall (which occurs when the negatively charged Balloon is near the Wall), and current location and relative speed (when in motion) of Balloon release. The amount of dynamic information that happens outside the user's keyboard focus that must be conveyed to the user is far beyond what might be conveyed by alerts in a typical web page.

The *Balloons and Static Electricity* sim underwent far more implementation iterations than the sims previously described. An example of an early implementation included: the Balloon implemented as a custom element using the ARIA application role with associated instructions on how to move it read out automatically, followed by the Remove Wall button, Two-balloon Experiments checkbox, and the Reset All button. The radio button group controlling charge views was not made accessible until later in the design process. In this early implementation, interaction with the Balloon involved moving focus to the Balloon and then hearing not only its label, "Yellow Balloon", but hearing its role "Application", its technical interactive state, "draggable", and finally a description of how to move it, "Grab and drag Balloon up, left, down, or right with the *W*, *A*, *S*, or *D* key." Once users moved the Balloon, they would hear alerts that would guide them towards an object (Sweater or Wall), simultaneously providing direction, progress, and actual location. They would hear the following five alerts as they progressed towards the sweater with successive presses of the *A* key: "Left. Towards sweater", "Left. Closer to sweater", "Left. On left side of Play Area", "Left. Near sweater", "Yellow Balloon has a few more negative charges than positive charges. Sweater has a few more positive charges than negative charges."

Challenges. These early implementation approaches resulted in multiple challenges, particularly for users with visual impairments. These challenges included difficulty recognizing the Balloon as the most important sim object, difficulty understanding how to interact with the Balloon, and difficulty recognizing the location of the Balloon during interaction.

When interactive HTML elements are accessed while using screen reader software, the user hears both the label of the interactive element (i.e., its accessible name) and the elements functional role. Ideally, the label describes what an element is or does in context, and the role reveals to the user how they will interact with the interactive object. Exploring an early prototype for the accessible *Balloons and Static Electricity* sim using only the *Tab* key (keypress indicated by *[Tab]*), would sound like the following, with some variation depending on screen reader software used: [*Tab*] "Application, Yellow Balloon, draggable", [*Tab*] "Remove Wall, button", [*Tab*] "Two-Balloon Experiments, checkbox, not checked" and [*Tab*] "Reset Balloon, button." If the user did not move on after hearing the label and role, further descriptive content (e.g., help text for how to move the Yellow Balloon) may be read out, depending on the screen reader software in use.

Initial implementations of the Balloon resulted in users not recognizing the Balloon as an important object to interact with, and not understanding how to interact with the Balloon. The interaction required users to hear the Yellow Balloon element's role and state first (i.e., "Application, draggable"), which does not indicate to the user how to interact with the Balloon. The more useful information, the help text indicating how to grab and move the Balloon, followed this role and state information. Most users chose to move focus to the next interactive element, rather than taking the actions described in the help text. The more inviting interaction, based on its description, was the "Two-Balloon Experiments" checkbox, resulting in most users selecting the checkbox before interacting with the Yellow Balloon. This choice is undesirable for multiple reasons as this selection made the sim screen and potential interactions more complex (with two Balloons to interact with) before users had a chance to explore and make sense of a simpler (single Balloon) scenario.

Once users successfully began interacting with the Yellow Balloon, users often ran into challenges understanding the location of the Balloon, due to the original structure of the location alerts. Initially, each keypress used to move the Balloon resulted in an indicator of the Balloon's direction of motion. For example, when repeatedly pressing the *A* key to move the Balloon to the left, the initial word of the alert read out was always the word "Left." If the user made successive soft keypresses (pressing lightly), the Balloon would not change in position enough to get a full change in location description, and would hear only the direction indicator, for example "Left. Left. Left.". If users made harder keys presses (advancing further) and also did so quickly, the alerts would interrupt themselves, reading out, "Left. …. Left. …. Left. …" In both cases, soft keypresses or quick successive key presses, users did not get a sense of progress or a sense of where the Balloon actually was in relation to the Sweater or Wall.

Solutions. To address the challenge of users having difficulty interacting with the Balloon, we implemented a two-step process for Balloon interaction. Users first “grab” the Balloon through selection of the “Grab Yellow Balloon” button, a native HTML interaction. Once the Balloon is “grabbed,” the user receives the information that it is an “Application” and the following help text “Press W, A, S, or D key to move Balloon up, left, down, or right”. While this two-step interaction has been more intuitive for non-visual users who access the description content, this interaction is less intuitive for those relying solely on the visual representations in the sim. We added visual cues (Fig. 6) that appear during initial interaction with the Balloon to provide visual guidance for this two-step interaction. With carefully designed auditory description as well as visual cues, the two-step interaction with the Balloon simultaneously supports visual and non-visual access.

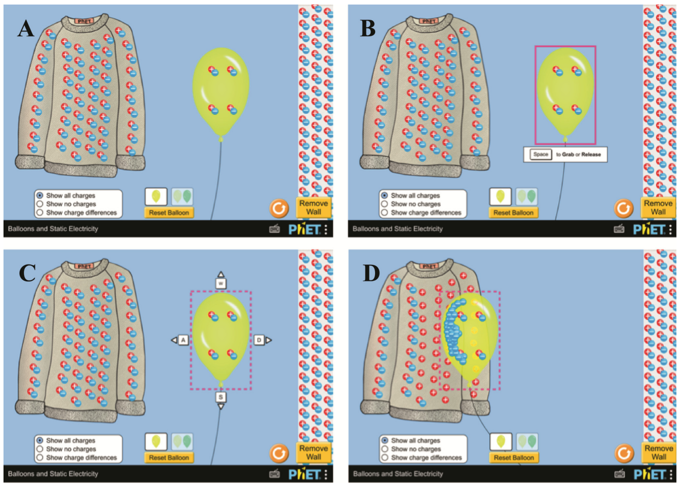

Fig. 6. Screenshots of initial interaction with Balloon in the PhET Simulation *Balloons and Static Electricity*. (A) Sim at opening, (B) focus on Balloon, not yet grabbed; (C) focus on Balloon, grabbed, showing letter and arrow key options for four-way movement; (D) moving Balloon on sweater (Color figure online).

To cue users with visual impairments that the Balloon is the most important object to interact with, we added two interaction hints to the PDOM description. The first is placed at the end of the Scene Summary that describes the sim on page load, “Grab balloon to play.”, and the second, “Look for grab button to play.” is placed at the end the Balloon’s description, just before the “Grab Yellow Balloon, button”.

To ensure users are kept up-to-date on the location of the Balloon during user-controlled motion and upon release, we implemented a sophisticated alert structure that still provides the user with directional information, a sense of progress and actual location, but does so more efficiently and ensures that the start of most alert text is unique to avoid repetition. After the Balloon is grabbed, a series of moving alerts that occur with successive presses of the *A* key now sounds like this, “Left. Closer to sweater”, “On left side of Play Area”, “On left side of Play Area”, “Near sweater”, “On sweater. Yellow Balloon picks up negative charges from sweater”, and finally, “Yellow Balloon has a few more negative charges than positive charges. Sweater has a few more positive charges than negative charges.”

4 Conclusions

In this work, we presented three design scenarios. For each scenario, we described our initial approach to implementation of accessibility features, challenges encountered, and the resulting – more successful – design. In our first example, we implemented sliders with all typical slider features, as well as enhanced features to support access to the specific pedagogical goals of the simulation *Resistance in a Wire*. The resulting sliders provide more efficient access to immediately useful numerical values than our initial implementation, while still providing access (through the use of a custom variable step size and a modifier key) to all possible input values. The resulting simulation provides improved access to the desired user interactions when utilizing keyboard input with or without screen reader software.

Our second example expands upon the use of sliders, this time implemented to provide access to unique arm and leg motion in the *John Travoltage* simulation. Use of sliders for this interaction provides direct access to the arm and leg via the keyboard, though required some refinement to determine the most effective input range, and to improve the descriptions of the location of the arm and the leg when accessed using screen reader software.

In our final example, we describe the design and implementation of a fully custom interaction to support four-way Balloon movement in the simulation *Balloons and Static Electricity*. In this design scenario, initial implementation presented significant usability challenges for visually impaired users utilizing screen reader software. To address this, we implemented a two-step process to first grab and then move the Balloon. This approach was significantly more successful for users of screen reader software, but introduced a new challenge for sighted keyboard users – as the two-step process was less intuitive without auditory descriptions. To address this challenge, we designed visual cues that appear during initial balloon interaction, resulting in similar guidance being provided for both visual and non-visual access. The result is a relatively intuitive user experience for a fully custom interaction.

4.1 Limitations

Throughout the design of these solutions, we incorporated analysis of user interviews and direct feedback from assistive technology users, as well as expert pedagogical and interface design expertise. This approach provided many insights that guided our designs, but is not an exhaustive representation of potential needs and preferences of the many users of keyboard and screen reader accessible content. We wholeheartedly acknowledge that more feedback from students, parents, and teachers will benefit our design approach, and may result in future improvements to the solutions described in this work. We expect to receive more feedback and to continue refining our design ideas and solutions once more accessible PhET simulations are publicly available and there is broader awareness and use of their accessibility-related features.

4.2 Looking Forward

In the presentation of these design scenarios, challenges encountered, and our solutions, we aim to contribute to the knowledge base for general strategies to the design and development of accessible interactive learning resources. These example scenarios were selected from the many design challenges we have encountered while implementing accessibility features into a subset of the PhET simulations. As we work with each new simulation, we add new strategies to our toolbox of approaches, and become better able to address challenges in PhET's more complex simulations. We welcome opportunities for further discussion, sharing of insights, and generalization of solutions with the broader accessibility community.

Acknowledgments. We would like to thank the PhET team, our collaborators at the Inclusive Design Research Centre at OCAD University and the Sonification Lab at the Georgia Institute of Technology, and research participants for their contributions to the design and development of the accessibility features for the three simulations discussed in this work. This material is based upon work supported by the William and Flora Hewlett Foundation, the University of Colorado Boulder, and the National Science Foundation under DRL-1503439 and DRL-1621363.

References

1. D'Angelo, C., Rutstein, D., Harrison, S., Bernard, R., Borokhovski, E., Haertel, G.: Simulations for STEM learning: systematic review and meta-analysis. Technical report, SRI International (2014)
2. Keane, K., Laverent, C.: Interactive scientific graphics recommended practices for verbal description. Technical report, Wolfram Research (2014)
3. Ellis, B., Ford-Williams, G., Graham, L., Grammenos, D., Hamilton, I., Lee, E., Manion, J., Westin, T.: Game accessibility guidelines (2017). http://gameaccessibilityguidelines.com
4. Smith, T.L., Lewis, C., Moore, E.B.: Description strategies to make an interactive science simulation accessible. J. Technol. Pers. Disabil. **5**, 225–238 (2017)
5. Smith, T.L., Lewis, C., Moore, E.B.: A balloon, a sweater, and a wall: developing design strategies for accessible user experiences with a science simulation. In: Antona, M., Stephanidis, C. (eds.) UAHCI 2016. LNCS, vol. 9739, pp. 147–158. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40238-3_15
6. PhET Interactive Simulations (2018). http://phet.colorado.edu/
7. Accessible PhET Interactive Simulations (2018). http://phet.colorado.edu/en/about/accessibility
8. Wieman, C., Adams, W.K., Perkins, K.K.: PhET: simulations that enhance learning. Science **322**, 682–683 (2008)

9. Perkins, K.K., Moore, E.B., Chasteen, S.V.: Examining the use of PhET interactive simulations in US college and high school classrooms. In: Proceedings of the 2014 Physics Education Research Conference, pp 207–210 (2015). https://doi.org/10.1119/perc.2014.pr.048
10. Sorge, V., Lee, M., Wilkinson, S.: End-to-end solution for accessible chemical diagrams. In: Proceedings of the 12th Web for all Conference, Article no. 6 (2015). https://doi.org/10.1145/2745555.2746667
11. Fitzpatrick, D.R., Godfrey, J.A., Sorge, V.: Producing accessible statistics diagrams in R. In: Proceedings of the 14th Web for all Conference, Article no. 22 (2017). https://doi.org/10.1145/3058555.3058564
12. Smith, T.L., Moore, E.B., Greenberg, J.: Parallel DOM architecture for accessible interactive simulations. In: proceedings of the 15th web for all conference (2018) (in Press)
13. King, M., Nurthen, J., Bijl, M., Cooper, M., Scheuhammer, J., Pappas, L., Schwerdtfeger, R.: WAI-ARIA authoring Practices 1.1 (2017). https://www.w3.org/TR/wai-aria-practices-1.1/
14. Faulkner, S., Eicholz, A., Leithead, T., Danilo, A., Moon, S.: HTML 5.2 (2017). https://www.w3.org/TR/html52/
15. Diggs, J., McCarron, S., Cooper, M., Schwerdtfeger, R., James Craig, J.: Accessible rich internet applications (WAI-ARIA) 1.1 (2017). https://www.w3.org/TR/wai-aria-1.1/
16. Moore, E.B., Smith, T.L., Randall, E.: Exploring the relationship between implicit scaffolding and inclusive design in interactive science simulations. In: Antona, M., Stephanidis, C. (eds.) UAHCI 2016. LNCS, vol. 9739, pp. 112–123. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40238-3_12
17. Hung, J.: PhET John Travoltage simulation design (2016). https://wiki.fluidproject.org/display/fluid/PhET+John+Travoltage+Simulation+Design