eCourse Accessibility Checklist

This document is intended to assist developers in the creation of accessible eCoures and provides specific guidance for working in Articulate Storyline 360 (SL 360)[1]. This document reflects but does not represent the UC Information Technology Accessibility Policy (pdf) or an official effort toward policy compliance. The standards herein reflect the WCAG 2.0 A/AA criteria adopted by the policy — in some cases translated to an eCourse-specific context — as well as select WCAG 2.1 criteria and other accessibility best practices.

This document was last updated: 6/27/2023.

General Accessibility

General accessibility standards pertinent to most digital environments, including eCourses.

Standards	Details
Program content in a way that conveys the same information, structure and relationships to assistive technology users	Any information, structure and/or relationships conveyed through visual presentation needs to programmed, or captured in text, in a way that enables assistive technologies to access and interpret the same information, structure and/or relationships[2]. Notes: • Laying out text in Storyline • Links and buttons in Storyline • Headings and heading hierarchies in Storyline • Tables in Storyline
Do not use color as the sole means of conveying information	If you want to use color to convey information, use it in combination with something else (e.g., text, icons, etc.) that conveys the same information.

[1] Some of this document's guidance remains relevant to Storyline 3 (SL3), as well.

[2] I.e., if possible in your authoring tool: program slide titles and other text that looks like headings as headings; program table structure; program lists as lists; links as links, buttons as buttons, etc.

Standards	Details
Avoid particularly dense or complex layouts	Use simple, well-organized designs with easily identifiable components and adequate spacing between elements.

Strive for consistency	Consistency helps users better understand and operate digital environments. Examples of helpful consistency include:
	• Consistent layout(s) and visual design(s)
	• Consistent reading order, tab order and/or Focus Order (or tab/focus order pattern[3]) in each slide
	• Consistent word/verb choice when describing actions in buttons, alt text and instructions. For example:
	  ○ "Explore ____ pop-up" for any button that opens a pop-up layer
	  ○ "Jump to ____ slide" for any button that directs users to another slide
	• Consistent programming approach

Proper reading order, tab order and Focus Order	Content should be sequenced in the order that best enables users to understand it and engage with it. Note: Reading order in Storyline.
Allow for magnification and style customization	Users should be able to: • Magnify text, so they can read it at larger sizes[4] • Modify certain style rules — such as font, font size, line height, etc. — with custom stylesheets or browser extensions Note: Storyline's Accessibility controls must be made available.

[3] I.e., even if the reading order/tab order/Focus Order are not exactly the same in all slides, because there are different elements in each slide, the same pattern or logic should govern the order in all slides.

[4] To allow for magnification in Storyline: if the Classic player is used, it must be locked at optimal size; if the Modern player is used, Zoom to fit must be toggled on (by default or by user choice).

Standards	Details
Help users know what to expect and do next	Use clear instructions, feedback, headings, labels and other approaches to help users understand what they encounter and need to do within an eCourse.
Use plain language	Strive to convey ideas clearly and concisely, with simple sentence structure that facilitates readability and comprehension. Explain any jargon, acronyms or abbreviations used. Explain, or avoid using, figures of speech, idioms, slang and other sayings that might not be understood by broad audiences.
Sufficient foreground/background contrast	Ensure there is sufficient contrast between any foreground content needing to be read and the background behind it. For text, adhere to the definitions and thresholds established by WCAG SC 1.4.3. Note: text can only be considered decorative (and exempt from this standard) if it can be rearranged or substituted without that changing its meaning or purpose. For graphical objects and user interface components that need to be read so they can be properly interpreted and/or operated (including the focus highlight), adhere to the threshold established by WCAG 2.1 SC 1.4.11 (3:1 — same as “large text”).
Use descriptive link and button text	Users should be able to understand a link or button’s unique purpose from the link or button text alone. Avoid using “Click here,” “learn more” or similarly generic or ambiguous phrases as link text. Avoid using URLs as link text. If a link directs to a file, include the file type in the link text: e.g., Word-to-PDF and PDF Accessibility Guide (pdf). Note: Button status in Storyline.

eCourse General Accessibility

General accessibility standards for eCourses.

Standards	Details
As universally accessible as possible, by default	An eCourse's default state should be its most universally accessible state. Users should have to actively choose settings that may create accessibility barriers. For example, by default: • Captions should start on; users can choose to turn them off (Turn captions on by default in Storyline) • Slide media should not "autoplay"; however, an "autoplay" setting can be provided for users to choose • Navigation should not occur automatically — that is, users should not be automatically advanced from one slide to the next when the slide's timeline ends; however, such a setting could be provided for users to choose • All activities should be fully accessible; inaccessible activities are avoided and never required for course completion; if an activity cannot be made fully accessible due to the nature of the activity, try to provide an accessible alternative within the eCourse[5] • In Storyline eCourses, it's recommended that accessible text be on by default (as described in the Accessibility controls section of this document)[6]

[5] For example, the original Stroop Color & Word Test requires both sight and the ability to differentiate colors, so it cannot be made fully accessible; however, it could be presented in an eCourse along with accessible alternatives, such as versions of the Stroop Test that utilize position, numbers or animals instead of colored words, or a version based entirely on audio information instead of visual information.

[6] However, if accessible text is turned on by default, be aware that the text positioning you see when designing a slide may not match the text positioning you see when the slide is previewed or published.

Standards	Details
Provide easy-to-access accessibility instructions (in addition to general user instructions)	Accessibility instructions should be provided within an eCourse and should be easy to access from anywhere within the eCourse. For example, you could have an Accessibility Instructions slide that can be accessed via: the slide menu, a link in the course player, a button that is available in all slides and/or a keyboard shortcut that is available in all slides. Accessibility instructions should also be provided before users open an eCourse. For instance, if an eCourse is hosted in an LMS, the eCourse's LMS-based course description[7] should include the accessibility instructions. General user instructions are also recommended.
Essential course content is available to assistive technologies	Screen readers and other assistive technologies must be able to access/read all essential content (e.g., all text, all controls). Users must be able to engage with and complete all aspects of an eCourse using only their keyboard. Use of a mouse/trackpad should not be required.
Hide decorative or duplicative content[8] from assistive technologies: aka, "accessibility tools"[9]	Assistive technology users should not have to encounter content that is decorative, duplicative or, in other ways, does not contribute to their learning experience in an eCourse. Note: Accessibility settings for individual elements in Storyline.

[7] In Storyline: this is accomplished through the Description field in the Publish panel. Use HTML to format Storyline course descriptions, but do not include breaks between tags.

[8] "Decorative content" is content that does not communicate information and/or is only intended to enhance visual design.

"Duplicative content" typically refers to images or other graphical elements that are intended to share information that is already being shared, in full, by adjacent, more accessible means, such as by adjacent, semantically-programmed text.

[9] "Accessibility tools" is the term Storyline uses to refer to assistive technologies.

Standards	Details

Prepare an alternate version	Prepare a fully accessible alternate version of the eCourse that can be deployed if an accommodation request is made.
	
	The alternate version should seek to provide an equivalent learner experience as the eCourse; this means, providing things like:
	
	• A transcript of each slide's audio
	• An alternative for time-based media for each video
	  ○ And preferably, an accessible means to access/view videos, and their audio descriptions (if applicable), outside of the eCourse (e.g., in YouTube)
	• All non-decorative slide content, including slide text and links
	• All links and files provided outside of slides, such as those provided through a Resources slide or menu
	• An opportunity to engage with quiz questions and other forms of interactivity; for example:
	  ○ Allow screen reader users to encounter all quiz question answer choices before being informed which answer are correct or incorrect, so they have an opportunity to try answering the question
	  ▪ You may even put the identified correct/incorrect answer choices on a separate page to provide more of an opportunity for other users to try answering the questions
	  ○ Identify the correct and incorrect answer choices and provide their corresponding feedback
	• Possibly even an equivalent means of demonstrating completion of the eCourse, such as an accessible comprehension quiz provided through a different format (e.g., a Google Form)

eCourse Interactivity and Navigation

Standards pertaining to interactivity and navigation within eCourses.

Standards	Details
Full use of eCourse controls	Assistive technology users must be able to engage with all interactive controls within an eCourse, including navigation controls and media controls. Notes: • Slide Menu in Storyline • Resources menu in Storyline
Default manual navigation[10]	By default, navigation within an eCourse — e.g., moving from one slide to another; opening and closing layers; etc. — should occur as a result of manually-input user commands, such as clicking a button, executing a keyboard shortcut or using a slide menu. Conversely, by default, layers should not open or close automatically[11]; nor should users be automatically advanced from one slide to the next when a slide's audio or timeline ends[12]. You can provide an automatic navigation option for users to choose within the eCourse, so long as users are presented with the choice before encountering automatic navigation and able to change their choice at any time while engaged with the eCourse.

[10] In Storyline: by default, slides should use the "Slide advances: By user" slide property.

[11] Except in situations where a layer is used to serve as the slide's "base layer"; e.g., if a slide has a "before and after" states, layers automatically opened through "show layer when slide timeline starts" triggers may serve as the base layer for each state.

[12] This behavior essentially imposes a time limit on how long users have to engage with the content in a slide or layer. Under WCAG SC 2.2.1, Storyline's pause/play slide controls are not sufficient to manage this time limit since those controls cannot be used before encountering the time limit: i.e., you'd only be able to pause a slide within its time limit. The behavior could also conflict with WCAG SC 2.2.2.

Standards	Details
Avoid time limits	Do not impose time limits on activities. If a time limit enhances the experience for certain users, provide it as a secondary option. If a time limit is essential to an activity, users should be informed of the time limit before encountering it and, ideally, be able to access accommodations within the eCourse itself, such as by having the option of choosing an untimed version of the activity and possibly as well, versions with longer time limits (e.g., two to three times the default time limit) or a user-assigned time limit.
Easy restart of slides and slide pop-ups	Users should be able to easily restart slides and slide pop-ups that contain audio, video or timed interactions from their beginning. Note: Restarting slides and pop-up layers in Storyline.
When a pop-up (layer or modal) opens…	Screen reader Focus should go to the first readable element in the pop-up[13]; users should not have to continue reading through base layer content in order to reach the pop-up content. Screen reader users should be notified that a pop-up has opened, such as through an audio cue or by having the first element in the layer automatically read by the screen reader[14]. Content outside a pop-up (e.g., content in the base layer or in other layers) should be hidden from assistive technologies while the pop-up is open[15], unless the content is necessary for a specific purpose.

[13] In Storyline: customize the slide's Focus Order so that the pop-up layer's content immediately follows the button used to open the pop-up layer; and/or, use the "Prevent the user from clicking on the other layers" slide layer property, which should also result in the first element being automatically read.

[14] In Storyline, this can be achieved by using the "Prevent users from clicking on the other layers" layer property, and/or by using a dialog layer.

[15] In Storyline: use the "Prevent the user from clicking on the other layers" slide layer property to hide base layer content from assistive technologies.

Standards	Details
Describe interactive functionality	Use button text, alt text or other means to describe interactive element functionality[16] (if it is not communicated by default). If there is text associated with an interactive element — e.g., a button label outside the button — consider if it should be incorporated into the interactive element's alt text and otherwise hidden from accessibility tools, so assistive technology users only have to read and engage with one element instead of two. If both the associated text and interactive element are made available to assistive technology users, provide a means of keeping track of which text relates to which interactive element[17]. If an interactive element has multiple states relating to meaningful engagement — e.g., selected/unselected, visited — each state's alt text should somehow convey which state is presently active. Describe activity-specific functionality in the activity instructions and global functionality in the eCourse instructions. Note: Button status in Storyline.
Provide keyboard shortcuts for common in-course actions	For example: a keyboard shortcut to jump to the next slide; a keyboard shortcut to jump to the previous slide; keyboard shortcuts to pause/play slides; keyboard shortcuts to jump to instructions slide(s); a keyboard shortcut to mute/unmute eCourse audio; a keyboard shortcut to access the slide transcript (if applicable); etc. Describe these shortcuts in the course/accessibility instructions, including any instructions provided before users open the eCourse. Note: Keyboard shortcuts in Storyline.

[16] E.g., "Currently unselected, True, press enter to select" for a button's unselected state; "Re-visit pop-up" for a visited state; assuming those states aren't communicated to assistive technologies by default.

[17] E.g., beginning the associated text with "Choice 1" and then using "Currently unselected, Choice 1, press space to select" as the alt text for the interactive element; "Choice 1" connects both elements.

Standards	Details
Do not use interactivity types that require sight and/or use of a mouse	For example, do not use: • Drag-and-drop interactions ○ Unless they can also be operated via selection functionality[18] by *both* keyboard-only users and screen reader users[19] • Hotspot interactions that lack pre-defined selection areas that are accessible to assistive technologies • Hover states to reveal important information or facilitate essential interactions

[18] Sarah Hodge provides an example of a drag-and-drop activity with built-in selection functionality in this How to Create an Accessible Drag-and-Drop Interaction in Storyline 360 tutorial. However, to be accessible by this Checklist's standards, the example in this tutorial needs to do a better job of conveying to screen reader users all of the information that is seen visually: for instance, the buttons in the first slide read exactly the same whether or not that "+1" icon is present; state-specific alt text is needed to convey which messages have been visited/completed and which haven't.

[19] In recent testing, it was found that Articulate Rise's drag-and-drop-based Sorting interactive block could be used by keyboard-only users via selection functionality; however, that selection functionality did not work for screen reader users, so that interactive block should not be used within Rise.

eCourse Multimedia Integration

Standards for integrating multimedia (graphics, audio, video and animated content) in eCourses.

Standards	Details
Provide descriptions of non-decorative visual elements	**Simple images/graphics:** assign succinct, descriptive alt text. **Graphs, tables, diagrams and other complex graphics:** assign succinct, descriptive alt text *and* relay all of the graphic's information through another means, such as through semantic structure or a separate, robust audio or text description. See the W3C recommendations for complex images. If a slide has audio, it is recommended that the audio describe important visuals and/or their key takeaways. Remember to hide decorative or duplicative images/graphics. This W3C alt text decision tree can help you determine what alt text, or other treatment, might be appropriate for an image/graphic.
Timed content should always be available to assistive technologies	Content that is programmed to appear/disappear at certain times during a slide should be available at all times to assistive technologies[20]. For example, if you have a text box that is visible when the slide starts and then disappears after 30 seconds: A) there should be a version of that text box that is always available to assistive technologies (a common strategy is to hide this version behind the slide's background); and B) the version that disappears after 30 seconds should be hidden from assistive technologies so they don't encounter two versions of the same content.

[20] Because you cannot anticipate when assistive technology users will choose to explore a slide's contents; screen reader users, in particular, may miss content that appears and disappears before they explore a slide's contents, especially in situations where their exploration of the slide is the only way for them to know the content was present (i.e., there is a cue to search the slide for the content).

Standards	Details
Avoid “strobe effects”	Minimize content that appears/disappears in rapid succession. Strobing, flickering or flashing effects can trigger seizures. If such content is unavoidable, provide a warning in advance.
Full use of media controls	Assistive technology users must be able to use all media controls within an eCourse, including controls for embedded media. All users should be able to pause/stop and resume media. Note: Media controls in Storyline.
Avoid autoplayed media (i.e., non-consensual media: media that plays without express user consent)	Audio and/or video should not play in a slide without the user’s consent, such as is exercised by clicking a play button or engaging in such a way that can reasonably result in media being played[21]. That is to say, users must choose/consent to play all media. Choosing to go to the next slide does not sufficiently constitute consent to automatically play audio or video in the next slide. Having to manually start each slide’s media might not be the ideal experience for many users, so it’s recommended you implement an “autoplay” option[22] that users can choose within the eCourse.

[21] For example, if you submit a quiz question answer choice for evaluation, you can reasonably expect to hear feedback for that answer choice, so it would be okay for the feedback audio to begin playing automatically after a user clicks a Submit button, without the user needing to click a separate “play feedback audio” button.

Similarly, if a user chooses to open a pop-up, they can reasonably expect to hear the pop-up’s audio, so again, a separate “play pop-up audio” button would not be necessary.

[22] That is, an option that causes slide audio or video to automatically play once a user arrives in the slide.

Standards	Details
Captions are required (and transcripts are recommended) for: • All videos • All slides that contain audio[23]	Videos and slides that contain audio must be captioned. Audio descriptions must be captioned as well. Transcripts can be provided even when captions are provided, as users may appreciate, and benefit from, being able to access information in that format. Assistive technologies should be able to read captions[24]. Assistive technologies *must* be able to read captions when they serve to translate a foreign language or, similarly, provide information that is otherwise not available to users. Notes: • Turn captions on by default in Storyline • Creating custom captions in Storyline • Transcripts in Storyline
Slide content should still be viewable while captions are displayed	Avoid positioning non-decorative content in slides in a way that would cause it to be hidden or obscured by captions[25].

[23] While it has been for many years a ubiquitous practice within eLearning to only provide a transcript to accompany eCourse slide audio, WCAG SC 1.2.2 actually requires that all eCourse slide audio be captioned, since it requires that all audio in "synchronized media" be captioned and an eCourse would technically be considered synchronized media; its slide audio is synchronized with another format of presenting information — slide text and other slide contents.

[24] Captions provided within a Storyline/eCourse slide are superior to captions provided within the video (file) itself, as the latter cannot be read by assistive technologies.

Storyline's built-in captions can be read by screen readers, but they are currently positioned at the very end of an eCourse's DOM order (i.e., the eCourse's reading/focus order, after the Resources menu).

[25] This is essential so that users who rely on captions can have an equivalent experience — a key accessibility principle — since users who do not rely on captions are able to receive the slide audio's information and view all the slide content simultaneously.

Standards	Details
Provide audio descriptions for videos, and other time-based, synchronized media, that contain information that is only shared visually	Non-decorative visual information in videos[26] must be relayed to users *while* the video plays. It is not sufficient to provide an alternate media description (example pdf) that is consumed separate from the video, though some users may nonetheless appreciate being offered one. Often, the simplest way to achieve this audio description standard is to have the video's audio describe all the visual information, but if that is not done, a separate audio description needs to be provided. The audio description can take the form of a secondary audio track that plays in sync with the video and describes important visual information during moments that lack important video audio[27]. The video may also be automatically paused and resumed as necessary to provide sufficient time amid the video's audio to describe important visuals through the audio description — aka, an "extended audio description" (see example linked further below). Audio descriptions, or extended audio descriptions, can also be provided in the form of a second version of the video that can be accessed alongside the original video (see example below). (Extended) Audio description in Storyline example.

[26] Remember, text can only be considered decorative if it can be re-arranged or replaced without that affecting the text's meaning or purpose.

[27] E.g., in between dialogue, voice-over audio, sound effects that are important to comprehension, etc.

Noted Storyline Features That Impact Accessibility

This is not an exhaustive list of accessibility-related Storyline features but rather a collection of less-publicized features that impact accessibility and pertain to this document's guidance.

Bugs, updates and testing

Storyline's accessibility features can be buggy — e.g., explore Focus unpredictability when hiding a layer — and accessibility-related behavior sometimes changes with Storyline updates[28], so it's imperative that you test what you build (early and often!) and adjust your development approach as necessary.

At minimum, test all aspects of a build with keyboard navigation, but for the best accessibility testing results, test with a screen reader and/or screen reader user.

Make sure to thoroughly test projects converted from earlier versions of Storyline, as these sometimes exhibit unique issues, like alt text changes not persisting. (Projects converted from earlier versions of Storyline also need Upgrade Project Text applied to them.)

Re-test after installing a Storyline update, and consider not installing updates mid-build.

Laying out text

Using Shift + Enter when laying out text in Storyline will create a paragraph break in the HTML output. This could constitute a WCAG failure and be a problem for assistive technologies, as it disrupts the semantic structure[29].

So, do not use Shift + Enter to achieve a certain layout[30]. Instead, use text box width or expanded spaces[31] to achieve the desired layout.

---

[28] Additionally, keep in mind that, due to these factors, as well as version differences, the Storyline behavior described in this document and in other resources may not perfectly match your experiences using and accessibility testing Storyline.

[29] E.g., for screen reader users, paragraphs may start or end mid-sentence, which could inhibit the screen reader user's ability to interpret the content properly (while also being a potential source of frustration).

[30] E.g., Do not use Shift + Enter to move certain parts of a heading or
sentence to another line of text, which is what was done to this sentence.

[31] Highlight the space between two words, open the Font settings, set the Spacing to Expanded and assign a value; though, be aware, expanded spaces may present differently with Accessible text.

Copy/paste

Storyline offers robust copy/paste capabilities, including the ability to copy/paste triggers, multiple triggers simultaneously, and layers and objects along with their trigger(s). Utilizing these copy/paste features can help you streamline many aspects of developing accessible content.

Classic player or Modern player?

Storyline offers two course players: Classic and Modern.

It's recommended that you use the Modern course player, as it offers numerous accessibility-related advantages over the Classic player:

- Better programming for assistive technologies
- Ability to assign two focus highlight colors, to ensure the focus highlight achieves contrast requirements against all backgrounds
- Ability to adjust caption font size independent of player font size
- Viewed slides are indicated by icons in the Menu instead of just by color

Maximizing accessibility with the Modern player

We currently recommend against using the following player features: Sidebar panes, Title and Logo (as Logo requires the presence of Sidebar content). If the Logo feature is used, give it alt text, as the feature that hides the Logo from accessibility tools does not work[32].

If you choose a custom color option for the course player, ensure it is one that allows for sufficient color contrast within the course player. Typically, this means avoiding colors that are toward the middle of the brightness/lightness spectrum, as such colors provide the least contrast with the auto-generated course player text and icon colors.

In the Player Properties panel's Colors & Effects tab:

- Assign light and dark Accessibility Focus Colors that provide sufficient contrast
- Assign font sizes that allow for player and caption text to be easily read
- Assign the Icon and text Button Style

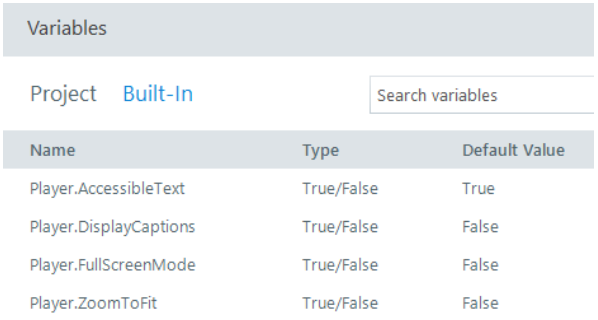
[32] If you uncheck "Visible to accessibility tools," screen readers will read the logo as "unlabeled graphic."

Accessibility controls must be made available to users

Accessibility controls is one of Storyline's player controls[33]. It allows users to toggle on/off various features that are essential to a fully accessible user experience and, for this reason, must be made available to users:

- Zoom to fit: only available in the Modern player; if toggled on, allows users to magnify slide contents
- Accessible text: if toggled on, text will be presented as HTML text instead of SVG text, which allows users to assign custom style rules to make text more readable and allows text to better interact with high-contrast display modes
    - It's recommended that accessible text be toggled on by default[34]
    - However, when accessible text is toggled on, the text positioning you see when designing a slide will not match the text positioning you see when the slide is previewed or published (examples of this are provided on the next page); the degree of positioning differences varies, depending on the specific font used[35]
- Keyboard shortcuts: if toggled on, allows users to use Storyline's built-in shortcuts

All of these features can be toggled on or off by default by changing the default value of the associated variable in the Variables panel's Built-In tab.



Accessible text can also be toggled on by default by first choosing Upgrade Project Text within the Font theme menu, then choosing Display Accessible Text by default (which will replace Upgrade Project Text in the Font them menu after you select Upgrade Project Text.)

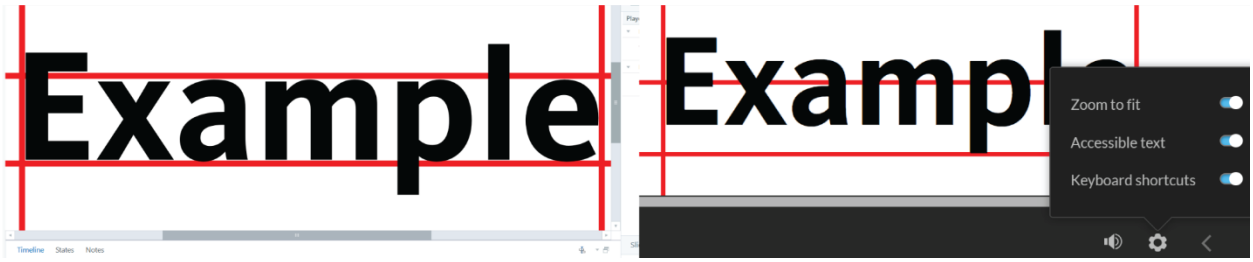
[33] Like Storyline's other player controls, it can be made available in an eCourse through the Features tab in the Player Properties panel.

[34] Toggling Accessible Text on by default ensures that the benefits provided by Accessible Text are available to users even if they're unaware of what Accessible Text is and/or how to toggle it on/off.

[35] These positioning differences can make designing in Storyline more challenging, as you may be forced to: A) repeatedly preview slides to explore how text will appear in the published output; and B) make text container sizes and margins different than what they would otherwise be. It's recommended that you choose which fonts you use based, in part, on which fonts present fewer positioning differences. For instance, as the examples on the next page illustrate one may want to use Arial instead of Kievit.

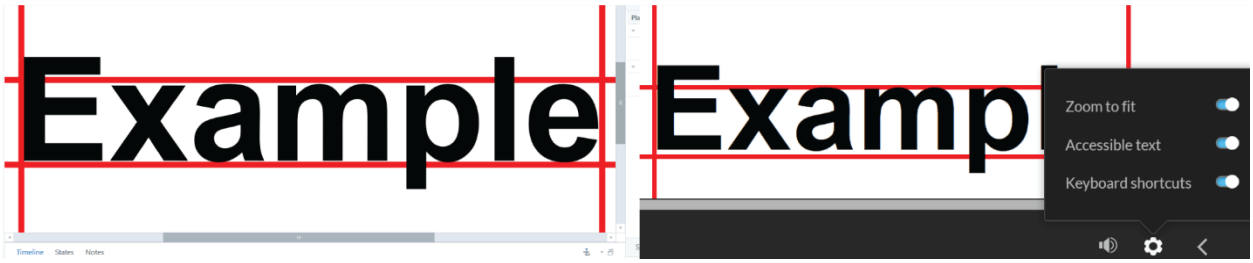
Example of text positioning differences when Accessible Text is toggled on

This first example uses the UCOP brand font Kievit.



Note the difference in text positioning when the slide is previewed, with Accessible Text toggled on: the text appears significantly (~25 pixels) higher than it did when designing the slide.

This second example uses the UCOP brand, recommended substitute font for Kievit: Arial.



Note the much less dramatic difference in text positioning, with Accessible Text toggled on: the text appears only slightly (~5 pixels) higher than it did when designing the slide.

Upgrade Project Text (a must if converting from SL3 to SL360)

If you are converting a project from SL3 to SL360 or wanting to use the new text autofit options, including the scroll bars option, you'll need to apply Upgrade Project Text. This feature is found at the bottom of the Fonts theme drop menu available in the Design ribbon and Slide Master view. Applying this feature may cause slight changes in font size throughout the project[36].

Additionally, certain issues — some of which may only present to assistive technologies[37] — have been reported to occur in SL 360 projects that were originally built in earlier Storyline versions and then converted to 360, so make sure to thoroughly test any converted project.

---

[36] It's recommended you apply this feature as early as possible in a conversion process, so you have more opportunities to notice and correct unwanted font changes.

[37] E.g., alt text changes that do not persist when the project is published or saved.

Bypassing the built-in manual start course feature

If the first slide of your course features media, the course will open in many browsers displaying just the course title and a play button. This title and play button do not present optimally to all assistive technologies, so consider bypassing them and creating a start-of-course experience of your own.

To bypass this feature, create a new first slide for your course that doesn't include media. Give the slide a jump to next slide when slide timeline starts trigger. You may also want to give the slide the same color as the course player and customize its player features so it goes unnoticed.

Links and buttons in Storyline

Links in Storyline

For links to be semantically recognized in Storyline eCourses, they must be created in a particular way:

1. Links can only be assigned to selected text, so start the process of creating a link by selecting the intended link text
2. Open the Insert ribbon and select the Hyperlink button (or use the Ctrl + K keyboard shortcut)

Assistive technologies will not be able to semantically recognize links that have been created by assigning "Open URL/file" triggers to elements; these, instead, will be recognized as buttons.

Buttons in Storyline

Any Storyline element that has a "when user clicks" trigger assigned to it will be semantically recognized as a button, even if the trigger has "Open URL/file" as its assigned action.

Transcripts

If you're providing slide transcripts within a Storyline eCourse[38], it's recommended that you use a properly constructed in-slide Transcripts method rather than using Storyline's built-in Notes sidebar.

---

[38] Remember: only captions are necessary to provide equivalent access to information conveyed via slide audio; transcripts are not necessary, but they can be provided to benefit users in other ways. Transcripts could be provided within an eCourse or they could be provided in the form of a linked accessible PDF or accessible HTML web page.

Reading/Focus order

Assistive technology reading/Focus Order in Storyline is controlled through the Focus Order panel. Order elements, in that panel, in the order you want them encountered by users.

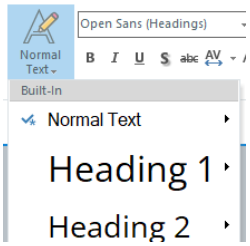
Be aware of how the "Prevent the user from clicking on the other layers" slide layer property affects how assistive technologies navigate a slide's reading/Focus Order.

If you experience an issue with focus not landing at the start of a text box or alt text description when a new layer opens[39], try this technique: Addressing focus not landing at start.

Focus unpredictability when hiding a layer

Storyline occasionally exhibits a bug where focus will not land in the appropriate spot in a slide's Focus Order when a layer that does not have the "Prevent the user from clicking on the other layers" layer property assigned to it is hidden. At present, no known solution for this bug has been found (other than assigning that layer property, if possible), but it may help to: A) "mention the bug in your course's screen reader instructions, to mitigate user confusion; B) program slide titles as headings.

Program slide titles as headings



Use the Text Styles menu in Storyline's Home ribbon to program headings.

Programming slide titles as headings is recommended for multiple reasons:

- It's appropriate (per WCAG SC 1.3.1) when slide titles are visually designed to look distinct from paragraph text, which is typically the case in eCourses
- It provides a helpful way for screen reader users to quickly return to a known spot within a slide's Focus Order (which is also often the start of the slide's Focus Order)

If the course title is present in the course player (via the Title feature), it will automatically be programmed as a Heading 1, making Heading 2 the appropriate heading level for slide titles. However, for reasons explained in the Headings and heading hierarchies section, we recommend against including the course title in the course player.

If the course title is not present, each slide title should be programmed as a Heading 1.

[39] This issue has been reported by NVDA users and replicated via NVDA testing but has not been reported or replicated (where it's known to exist for NVDA users) by JAWS users.

Dialog layers

In 2022, Storyline 360 introduced a “Present as: Dialog” slide layer property. If assigned to a layer, users will only be able to interact with that layer’s contents and will not be able to interact with anything else, including the course player, while that layer is open.

If a label and/or description are assigned, screen readers will automatically read those elements when the layer first opens; however, this automatic reading has been found to present differently with different assistive technologies, including some sub-optimal behavior.

Due to the above-described characteristics, thoroughly consider and test each use of this property and these features, and consider just using the “Prevent the user from clicking on the other layers” layer property instead.

Restarting a slide or pop-up layer

Storyline has two built-in methods that can, if set up properly, replay a slide or pop-up layer:

1. The Replay button that is available in the player controls when the Seekbar is enabled
2. The built-in “Replay the slide” keyboard short: Ctrl + Alt + R

Both methods trigger “When revisiting” functionality; if the “When revisiting” property is set to “Reset to initial state,” the slide/layer will restart from its beginning; if the “When revisiting” property is set to “Resume saved state,” the Replay button and shortcut won’t have any effect. It is generally recommended that you not use the “Automatically decide” setting for this property, since it’s behavior can be unpredictable.

Keyboard shortcuts

Storyline now comes with a collection of built-in keyboard shortcuts covering many common course actions. These shortcuts do not require screen reader users to toggle their screen reader commands off[40] in order to be used.

You can create additional, custom keyboard shortcuts through “When user presses a key” triggers. If you use Ctrl + Alt in the key press combination — e.g., Ctrl + Alt + A, Ctrl + Alt + L, etc. — screen reader users will not need to toggle their screen reader commands off in order to use the shortcut.

[40] Different screen readers have different terminology for this. In JAWS, you toggle screen reader commands off by toggling off the Virtual PC Cursor (Insert + Z). In NVDA, you toggle screen reader commands off by toggling from Browse mode to Focus mode (NVDA key + Space).

Prevent the user from clicking on the other layers

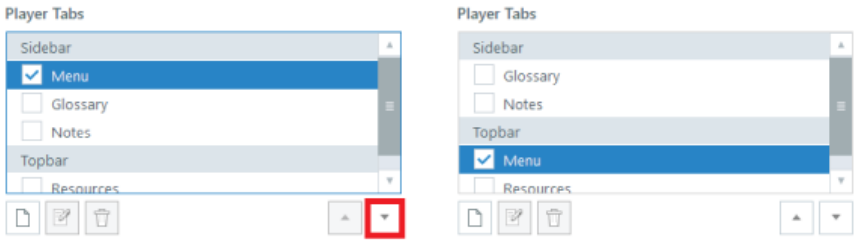
This is a slide layer property. If it is assigned to a layer:

A. When the layer opens, screen reader focus should go to the element in the layer that is highest in the slide's Focus Order, and that element should automatically be read
    a. Note: a bug may inhibit this functionality and cause focus to land elsewhere within a layer; the Addressing focus not landing at start section shares a method for addressing this bug
B. Assistive technology users will not be able to read or interact with elements in any other layer, including the base layer
C. When the layer is hidden, screen reader focus should return to the element that was used to show/open the layer (if applicable)
    a. This behavior may vary depending on the version of Storyline used to publish the project, as well as on the user's screen reader and browser
    b. If the layer was shown/opened via a key press trigger, upon hiding/closing the layer, screen reader focus may return to the element that had screen reader focus at the time the layer was shown/opened or go to the element just before that element in the slide's Focus Order

Slide Menu

Users must be able to complete a course and access all the course's content without having to use the course's slide Menu (i.e., the slide Menu is a supplemental navigation option).

It's recommended that developers use Storyline's built-in Topbar slide Menu instead of using Storyline's built-in Sidebar slide Menu.



Storyline's built-in slide Menu can be moved from the Sidebar to the Topbar by selecting it within the Player Properties panel (Features tab), then pressing the Player Tabs' down arrow button until Menu has moved from the Sidebar section to the Topbar section.

In-slide slide Menu

An in-slide slide Menu could also be used as an alternative to Storyline's built-in Sidebar slide Menu. This method may produce a preferable assistive technology experience for some users, while other assistive technology users may prefer the experience provided by Storyline's built-in Topbar slide menu.

Using Storyline's built-in Topbar slide Menu is likely preferable for eCourse developers, as it requires much less work to produce and maintain; which is to say, by choosing to use an in-slide slide Menu, you adopt a greater production and maintenance burden.

See In-slide slide Menu for guidance on constructing custom-built, in-slide slide Menus.

Accessibility settings for individual elements

In Storyline, the accessibility tool visibility[41] and alt text of individual elements are controlled through the Accessibility tab in the Size and Position panel[42].

There are a few quirks about these accessibility settings worth knowing; that is, if the accessibility settings you believe you've established don't match what you find when you test with assistive technologies, check these factors:

1. Elements can have different accessibility settings in different States
    a. Elements can even have different accessibility settings for their Normal state (when viewed through the Edit States mode) than they have outside of Edit States mode
2. Elements within another element's State can have their own accessibility settings
3. Groups have their own accessibility settings, separate from the accessibility settings of each individual element within the group
    a. Being in a group does not negate an element's accessibility settings

[41] That is, whether an element is visible to accessibility tools or not; assistive technologies will not be able to detect, read or interact with elements that are not visible to accessibility tools.

[42] To open this panel, right-click on an element and choose the Size and Position menu option or select an element and use the Shift + Ctrl + Enter keyboard shortcut.

Headings and heading hierarchies

If the course title is presented in the Player, it will automatically:

A) Be programmed as a Heading 1

B) Have a set position in the course's reading order that cannot (currently) be customized and will always come after the slide region's contents (i.e., the in-slide content):

- If a left sidebar is present in the Player, the title will follow it in the reading order
- If a left sidebar is not present in the Player, the title will follow the Prev/Next buttons and/or other player controls (e.g., seekbar, volume, etc.), depending on which are present

Since (A) and (B) combine to prevent you from providing a proper heading hierarchy when the slide title is present, it's recommended that:

- The course title not be presented in the Player
- Slide titles be programmed as Heading 1s

If the course title is present in the Player, consider programming slide titles as Heading 2s instead, since that would then be the appropriate heading level even though, technically, the course title's Heading 1 should come before the slide title's Heading 2 in the reading order.

Headings in layers

If a layer has the "Prevent the user from clicking on the base layer" layer property assigned to it, the layer should contain a heading that:

- Is the highest layer element in the slide's Focus Order
- Is programmed as a heading of the same level as the slide titles

Resources Menu

If you use Storyline's Resources Topbar drop menu (controlled through the Player Properties panel) to present links to various resources, you should provide all of those same links within eCourse slides, such as in the slide in which a resource was initially discussed and/or in a Resources slide that offers all of the same links presented in the Resources drop menu.

This approach helps ensure that all users are able to access the linked resources even if they have difficulty understanding and/or operating the Resources drop menu.

Tables in Storyline

Do not use tables for layout purposes (i.e., for the purpose of achieving a specific visual design).

Additionally, be aware that Storyline is not currently capable of programming table structure.

When screen reader users encounter a table that has been inserted into a Storyline slide, “table with ___ columns and ___ rows,” will be read out to them, but the table will not actually be programmed as a table, meaning screen reader users cannot:

- Find the table using jump to next/previous table screen reader commands
- Navigate the table using table-specific screen reader commands
- Have table header cell information available to them when they read a table data cell

If you include a table in a Storyline slide, it’s recommended that you also provide in the same slide a link to a properly programmed HTML (better) or PDF version of the table:

- (HTML) Creating Accessible Tables on WebAim.org
- (PDF, pages 31-36) HHS Section 508 Guide Tagging PDF’s in Adobe Acrobat Pro (pdf)

Table Content Reading Order in Storyline

By default, table content will be read left-to-right, top-to-bottom; however, each cell can be manually repositioned within the Focus Order panel to create a custom reading order.

Alt text changes not persisting

Another bug that has been known to occur, especially with projects converted from earlier versions of Storyline, is alt text changes not persisting; as in, you make an alt text change, publish or save the project, then test or re-open the project and find the alt text has reverted to what it was before. If this occurs, you may need to recreate the element from scratch.

Button status

Storyline currently has limited abilities to programmatically communicate button status to screen reader users: the built-in Disabled state communicates that a button is unavailable; radio buttons and check boxes currently communicate selected (“checked”) and unselected (“not checked”) statuses.

You will otherwise need to use state-specific alt text to communicate a button’s status.

Media controls

Test the controls for embedded media to ensure they can be used by all assistive technology users (e.g., keyboard navigators and screen reader users)[43].

The seekbar and play/pause course player controls can be used to control media *if* the media is synched to the slide timeline and always remains so[44].

[43] Storyline's Video Controls should now be accessible to all assistive technologies, but test nonetheless.

[44] One of the challenges in this regard is ensuring users cannot click on videos — either with their mouse cursor or through keyboard commands — since clicking on a video pauses it without also pausing the slide's seekbar. To prevent this, it's recommended that you: 1) hide videos from accessibility tools, so assistive technology users cannot click on them via keyboard commands; 2) place a rectangle with transparent fill over each video, so mouse users cannot click on them; 3) allow those rectangles to be visible to accessibility tools and give them the alt text you would otherwise give the video.

Storyline Techniques

Addressing focus not landing at start

As mentioned elsewhere, if you open/show a layer that uses the "Prevent the user from clicking on the base layer" layer property, focus will go to the element in that layer highest within the slide's Focus Order; however, NVDA users[45] may find that focus lands not at the start of the element but somewhere within it[46].

While the root cause of this issue is not fully known, it seems to primarily occur with text frames and alt text descriptions that exceed the screen reader's characters per line limit.

A simple solution is to ensure that the element in the layer that is highest within the slide's Focus Order does not exceed the NVDA's default characters-per-line limit of 100. For example, if the highest element within the layer is a text frame containing the layer's title and a few paragraphs of text, put the layer title in its own text frame and make that text frame the element within that layer positioned highest in the slide's Focus Order.

Avoiding autoplayed media

There are different ways you can achieve this standard within Storyline; however, achieving this standard typically isn't as simple as just requiring users to manually start media. There are often other factors you have to account for, such as:

- Keeping the media tied to the slide's timeline/seekbar so that they start and end together, and so users can reliably scrub forwards and backwards through the media using the seekbar in the player controls
- Accounting for users starting a slide using the Player's play button or the built-in, play keyboard shortcut
- Having the media continue, pause or stop when other layers open
- Having the media behave as desired when users revisit and/or replay the slide
- Allowing users to choose an "autoplay" option

So whatever method you choose to use, make sure you thoroughly test it.

---

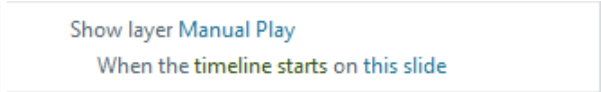
[45] This issue has only been reported and replicated by NVDA users but not JAWS users.

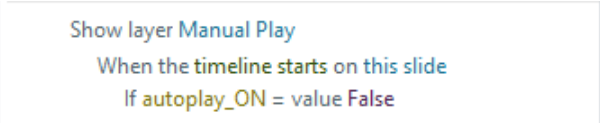
[46] E.g., not at the start of a text frame but rather a few sentences into the text frame.

Manual play layer

An effective method for avoiding autoplayed media and accounting for all the factors noted in the prior section is to use a “manual play” layer.

This layer will be triggered to show when a slide’s timeline starts; this trigger may include an “autoplay” True/False condition, as described in the Allow for an “autoplay” option section.





1. Give this layer the following layer properties:
    a. Hide other slide layers: optional
    b. Hide objects on base layer:
        i. If checked, no users will be able to view or read base layer content
        ii. If unchecked:
            1. Users will be able to see base layer content
            2. Screen reader users may be able to read base layer content depending on the “Prevent the user from clicking on the base layer” setting
    c. Hide slide layer when timeline finishes: **unchecked**
    d. Allow seeking: **no**
    e. Prevent the user from clicking on the base layer/other layers:
        i. If checked, no users will be able to interact with base layer content
        ii. If checked, the slide contents must be obscured from view — such as through use of a semi-opaque shape covering the layer’s background — so that all users have an equivalent experience in the slide while the layer is open[47]

[47] I.e., if screen reader users cannot read base layer content while this layer is open due to this layer property, other users must be prevented from being able to read the base layer content as well.

iii. If unchecked:

1. Screen reader users will be able to read base layer/other layer content and interact with it, as long as other triggers or settings aren't applied to prevent interactions

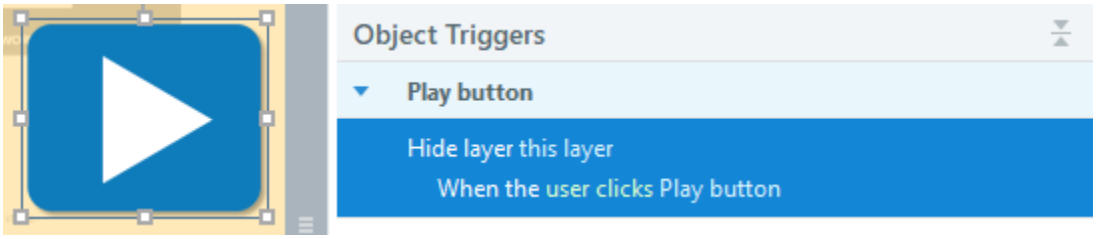
    a. However, if screen reader users are able to open a layer that has the "Hide other slide layers" layer property assigned, you may need to incorporate additional conditions and triggers in order to do so, as described in the Account for Hide Other Slide Layers section

2. Non-screen reader users need to have an equivalent experience as screen reader users; so, anything screen reader users can read and interact with, non-screen reader users need to be able to read and interact with as well

f. Pause timeline of base layer: **checked**

g. When revisiting: **Reset to initial state**

2. Include in this layer a button that users will click in order to play the slide's media
3. Give that button a hide layer, this layer, when user clicks this button trigger



a. When the layer closes, its "pause timeline of base layer" property will no longer be in effect, causing the base layer's timeline to start, allowing media in the base layer to play

4. Finally, in the slide's base layer, add a trigger that will hide the manual play layer when the slide timeline reaches time 0.01 seconds
    a. This trigger is necessary to account for users choosing to play the slide by using the play button in the player controls or by using the built-in, play keyboard shortcut
    b. Make sure this trigger is ordered beneath the trigger that shows the manual play layer when the slide's timeline starts[48]

Allow for an "autoplay" option

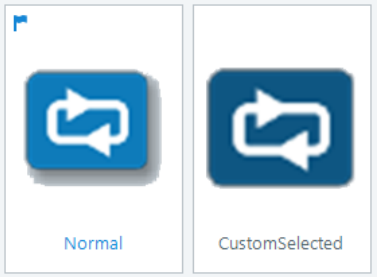
While the "manual play" setting should be a course's default setting, because by default, courses should be as accessible as possible, some users may not like having to manually play each slide's media and may prefer an "autoplay" option, which would allow slides and their media to play automatically as soon as one navigates to a slide.

To enable an autoplay option:

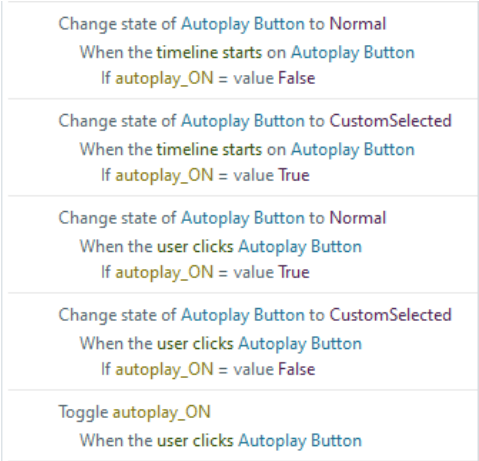
1. Create a True/False variable, with a default value of False, that you'll use to track if a user has chosen the autoplay option or not; for the sake of these instructions, let's call that variable "autoplay_on"
2. Edit the trigger that shows the manual play layer when the slide timeline starts and add to it the condition: if autoplay_on = value False

[48] Storyline 360 should ensure this order by default and should not allow any "when timeline reaches" triggers to be ordered above "when timeline starts" triggers.

3. Next, create a button that has a normal state and a custom state[49]; the two states should be visibly different so users can tell which state the button is in



4. Give the button's Normal state alt text like, "Toggle autoplay on, autoplay is currently off" or just "Toggle autoplay on"
5. Give the button's custom state alt text like, "Toggle autoplay off, autoplay is currently on" or just "Toggle autoplay off"
6. Give the button the following conditional triggers:



   a. Change state of this button to Normal when timeline starts on this button if autoplay_on = value False
   b. Change state of this button to custom when timeline starts on this button if autoplay_on = value True
   c. Change state of this button to Normal when user clicks on this button if autoplay_on = value True
   d. Change state of this button to custom when user clicks on this button if autoplay_on = value False
   e. (Adjust variable) Toggle autoplay_on when users clicks on this button

Account for "Hide Other Slide Layers"

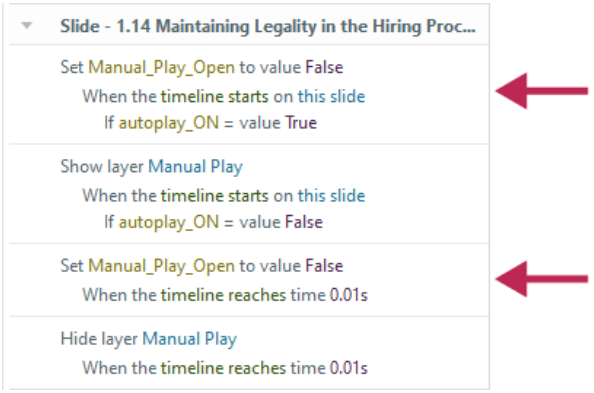
If you allow screen reader users to read and interact with the base layer while the manual play layer is open *AND* users are able to open another layer from the base layer that has the "Hide other slide layers" layer property, you will need to include additional triggers and conditions to ensure media on the base layer does not begin playing when users open that other layer (since opening a layer with the "Hide other slide layers" property would hide the manual play layer, thus unpausing the base layer's timeline).

---

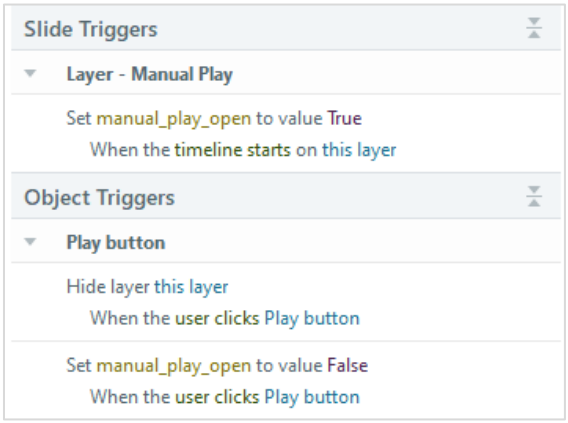
[49] Alternatively, you could use the built-in Selected state instead of a custom state, in which case you would not need triggers 6c & 6d, listed on this next page; however, there has historically been enough unpredictability with using the Selected state for this purpose, that it may not be a superior, or even advisable, option. Bottom line: if you use the Selected state, test it thoroughly.

In such cases:

1. Create a True/False variable, with a default value of False, that you'll use to track if the manual play layer is open; for the sake of these instructions, let's call that variable "manual_play_open"
2. On the slide's base layer, create an adjust variable trigger that will set manual_play_open to value False when the timeline starts on the slide; if you've allowed for an autoplay option, include an if autoplay_on = value True condition
3. Order this trigger above the trigger the shows the manual play layer when the slide timeline starts
4. Create another adjust variable base layer trigger that will set manual_play_open to value False when the timeline reaches time 0.01 seconds
5. Order this second trigger above the trigger that hides the manual play layer when the timeline reaches time 0.01 seconds

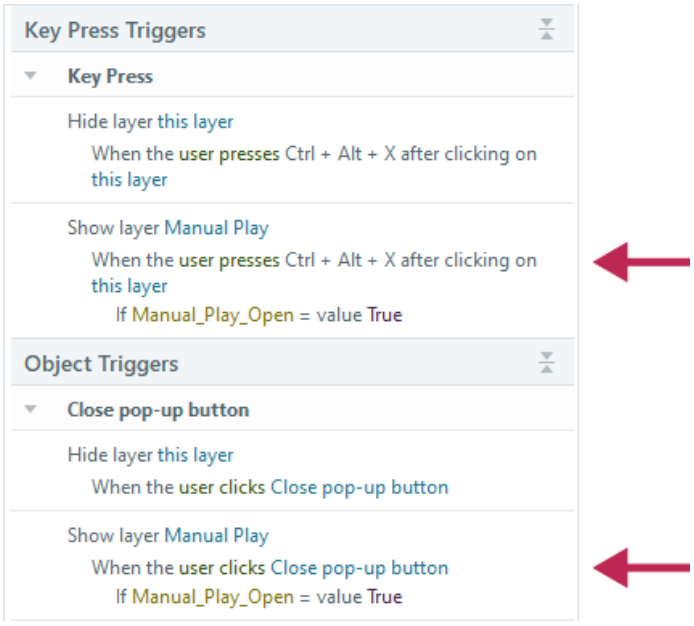


6. Add two triggers within the slide's manual play layer:
    a. (Adjust variable) Set manual_play_open to value True when timeline starts on this layer
    b. This second trigger will be associated with the layer's play button: (Adjust variable) Set manual_play_open to value False when the user clicks play button



7. Then, on the layer with the “Hide other slide layers” property:
    a. If you want the base layer’s media to always be paused when users open the layer, assign the layer the “Pause timeline of base layer” layer property, or...
    b. If you want users to be able to play the base layer’s media, then open this layer and have the base layer’s media continue to play, add the following trigger: pause timeline on *slide’s title* (which is how the base layer will be labeled) when the timeline starts on this layer if manual_play_open = True[50]

8. If there is a key press trigger in this layer that hides this layer, create an additional trigger that will show the manual play layer, when users presses the same keys, manual_play_open = value True
9. If there are any objects in this layer that have a hide this layer when user clicks trigger, add to them an additional trigger that will show the manual play layer, when user clicks on this object, if manual_play_open = value True



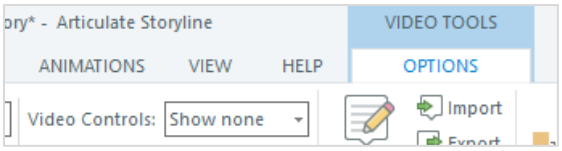
[50] In effect, this trigger uses the manual_play_open trigger to evaluate if users have chosen to play the base layer media or not, and if they have, manual_play_open will be set to False by other triggers, so this trigger will not fire when the layer opens.

Creating custom captions

If you want even more control over how captions appear and are read by assistive technologies[51], you can create your own custom captions.

For the following method to work, you need to keep the media's timeline connected to the slide's timeline, which means:

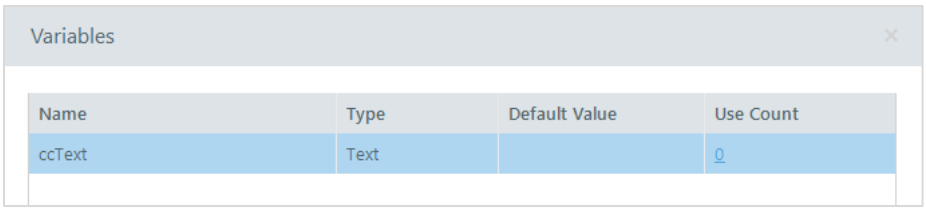
- The media needs to play automatically when the slide timeline is not paused
- The media cannot be paused independent of the slide timeline, so for video:
    - You need to set the "Video Controls" (in the Video Tools Options ribbon[52]) to "Show none"



    - You need to hide the video from accessibility tools so assistive technology users cannot keyboard navigate to the video and press Enter/Space while focus is on it, thus pausing the video while the slide timeline continues to run...
    - And you need to place a rectangle with solid fill and 100% Transparency over the entire video (i.e., give the rectangle the same size and position as the video) so that mouse users cannot click on the video, thus pausing it while the slide timeline continues to run
        - It's recommended that you allow this rectangle to be available to accessibility tools and give it whatever alt text you would give to the video

Then, to create the custom captions:

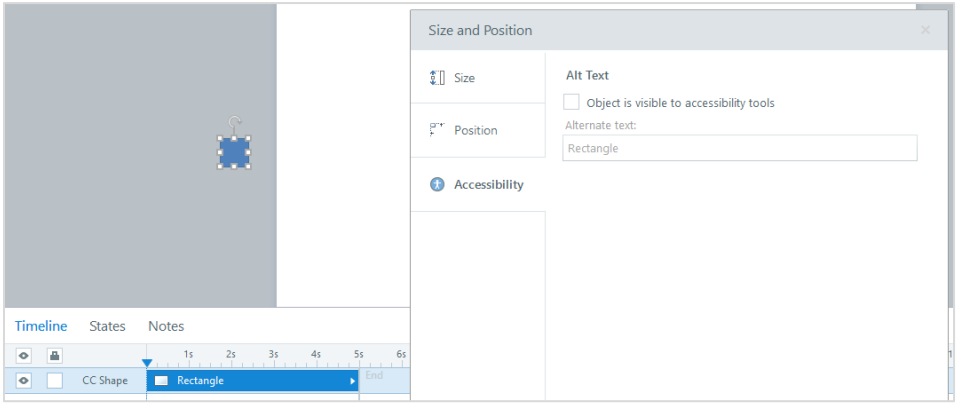
1. Create a Text variable with no default value; this will be the "caption variable"; for the sake of these instructions, let's call that variable "ccText"



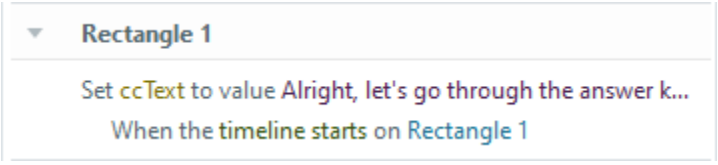
[51] As of this update, Storyline's built-in captions can be read, but they default to being the very last thing read in the eCourse environment (as in, a screen reader user would have to read through all the slide and course player contents before they reached the captions).

[52] Available only when a video is currently selected.

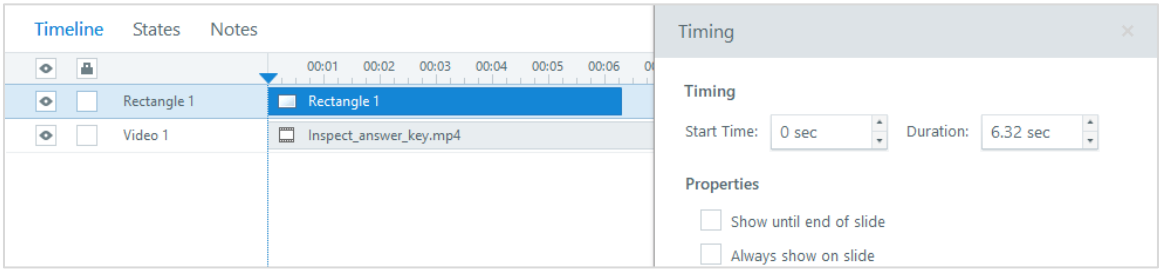
2. Create a shape, position it outside the slide's visible area and hide it from accessibility tools



3. Create a trigger associated with the shape that will adjust the ccText variable's value when the timeline of the shape starts, with the newly assigned variable value being the first segment of caption text[53]



4. Position the shape and adjust its duration within the slide's timeline so that it is present during/synched with the stretch of audio that corresponds with the caption text[54]

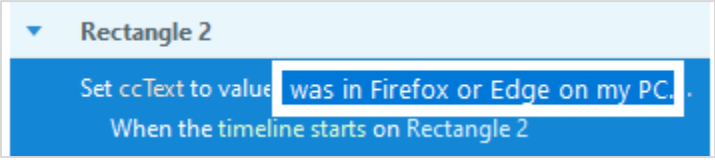


5. Next, copy the shape you created and paste another instance of it outside of the slide's visible area

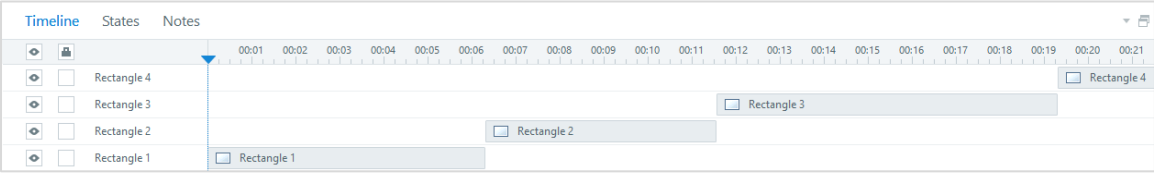
[53] Beware: the variable Value field will not accept paragraph breaks. If you copy a text segment that contains a paragraph break and attempt to paste it into the Value field, only the text before the paragraph break will be pasted in the field.

[54] It's possible to do this by manipulating the shape's presence within the timeline, but for the most accurate synchronization, it's recommended that you use the Timing panel, accessed by right-clicking on the shape within the Timeline and selecting "Timing…".

6. In the Triggers pane, adjust the variable value for the copied instance's trigger so that the value captures the next segment of caption text



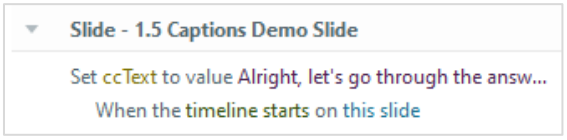
7. Position the new shape and adjust its duration within the slide's timeline so that it is present during/synched with the stretch of audio that corresponds with the caption text
8. Repeat this process as many times as needed so that all audio is captioned



9. Create a text box — let's call it "CC Box" — and include in it a Reference to the caption text variable you created
    a. To add a variable reference into a text box, place your cursor within the text box and use the Insert > Reference feature or type the variable's name within the text box with a "%" symbol immediately before and after it: e.g., %ccText%
10. Give this text box the size, position, shape fill, internal margins and text properties you desire for the caption text

Download video alternative description

11. Lastly, if the slide's "When revisiting" property is set to "Reset to initial state," create a trigger that will adjust the ccText variable's value to the first caption segment when the timeline starts on the slide, and make sure that trigger is positioned high enough in the slide's trigger order that it won't be impeded by another trigger[55]

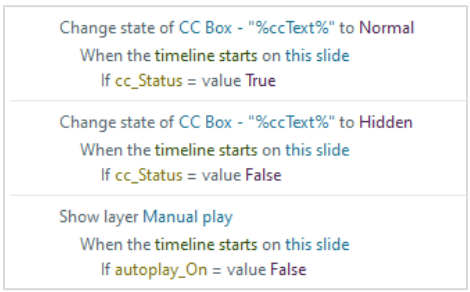


If these are intended to be open captions[56] your work is done.

Turning these into closed captions

Closed captions are captions that can be turned off and on. Since we want our eCourses to be as accessible as possible by default, closed captions should start on, requiring users to turn them off if captions are not wanted.

1. Create a new True/False variable with a default value of True; for the sake of these instructions, let's call this variable "cc_Status"
2. Create two new conditional slide triggers that will control CC Box's visibility at slide start:
    a. Change state of CC Box to Normal when timeline starts on this slide if cc_Status = value True
    b. Change state of CC Box to Hidden when timeline starts on this slide if cc_Status = value False



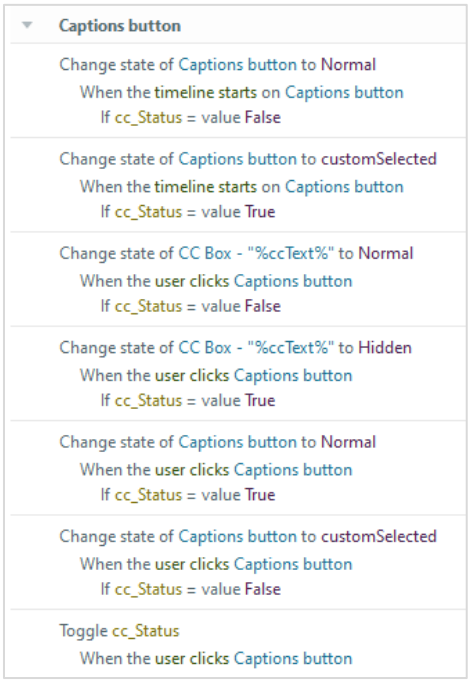
    c. If the slide includes a manual play layer, order these two triggers above the trigger that shows the manual play layer when the slide timeline starts
3. Create a button that will be used to turn the closed captions off and on
4. Give the button a custom state visibly different than its Normal state that will represent the button's "captions on" status
    a. Give the button's Normal state alt text like, "Toggle closed captions on"
    b. Give the button's custom state alt text like, "Toggle closed captions off"

---

[55] Such as the show manual play layer trigger when timeline starts if autoplay_on = value True trigger, which could pause the base layer before other triggers have a chance to be activated.

[56] Captions that are always visible and cannot be hidden or turned off.

5. Give the button the following conditional triggers:



   a. Change state of this button to Normal when timeline starts on this button if cc_Status = value False
   b. Change state of this button to custom when timeline starts on this button if cc_Status = value True
   c. Change state of CC Box to Normal when user clicks on this button if cc_Status = value False
   d. Change state of CC Box to Hidden when user clicks on this button if cc_Status = value True
   e. Change state of this button to Normal when user clicks on this button if cc_Status = value True
   f. Change state of this button to custom when user clicks on this button if cc_Status = value False
   g. (Adjust variable) Toggle cc_Status when users clicks on this button

Turn captions on by default

If you are using Storyline's built-in captions, you can have them turned on by default (for guidance on turning custom captions on be default, visit the previous section: Turning these into closed captions).

Open the Variables panel and select the Built-In tab. Scroll down the list of built-in variables until you find a variable named Player.DisplayCaptions. It will be a True/False variable set to False by default. Changing the default value to True will enable captions being displayed by default in that Storyline project.

Providing an automatic navigation option

By default, Storyline slides should be set to Slide advances: By user so that navigation does not occur automatically when slide timelines end; however, you could provide an option within the eCourse that would allow users to choose automatic navigation.

There are many ways this could be accomplished. Here is a simple method you could try:

1. Ensure every slide is set to Slide advances: By user
2. Create a True/False variable with the default value of False
    a. These instructions will refer to that variable by the name “autonav”
3. Create a button that users can use to toggle automatic navigation on/off and give it:
    a. Two visually distinct states (one of which could just be the button’s built-in Normal state[57]):
        i. A state associated with automatic navigation currently being off but available to toggle on (referred to hereafter as the “toggle on” state), with alt text like, “Toggle automatic navigation on”
        ii. A state associated with automatic navigation currently being on but available to toggle off (referred to hereafter as the “toggle off” state), with alt text like, “Toggle automatic navigation off”
    b. The following triggers:
        i. Change the button state to the toggle on state when the button’s timeline starts if the autonav variable equals False
        ii. Change the button state to the toggle off state when the button’s timeline starts if the autonav variable equals True
        iii. Change the button state to the toggle off state when a user clicks the button if the autonav variable equals False
        iv. Change the button state tot the toggle on state when a user clicks the button if the autonav variable equals True
        v. Toggle the autonav variable when a user clicks the button

[57] It’s recommended that you create a custom state for the button’s other state, as opposed to using Storyline’s built-in Selected state.

4. Create, and copy/paste into each applicable slide[58], a trigger that will jump users to the next slide when the slide timeline ends if the autonav variable equals True
    a. If you include in all slides the button that users can use to toggle automatic navigation on/off, you can attach this trigger to the button so you are only copy/pasting one thing — the button — into each slide instead of two things — the button and the trigger; if you go this route, have the trigger's When event be the button's timeline ending instead of the slide's timeline ending and ensure that the button's Timing setting "Show until end of slide" is toggled on[59]

However you choose to implement this optional functionality, ensure that:

- It is explained to users
- Users do not encounter automatic navigation before actively toggling it on
- Automatic navigation is never toggled on/off outside the user's direct control
- Users can change their automatic navigation choice at any time
    - They don't need to be able to toggle automatic navigation on/off in each slide, but they should be able to access that ability from any slide[60]
- Users' must have at least one way to change their automatic navigation choice that is not affected by their current automatic navigation choice
    - For example, if automatic navigation can only be toggled on/off in a single slide, that slide should not have the automatic navigation behavior because if it did and automatic navigation was currently toggled on, users may be automatically navigated away from the slide before they had the opportunity to toggle automatic navigation off

[58] As in, automatically progressing users to the next slide may not be appropriate for all slides, depending on their nature, even when the automatic navigation option is toggled on. For example, if a slide contains elements — like links, pop-ups, etc. — that you want or need learners to engage with before proceeding, automatically navigating users past that slide when its timeline ends may not be appropriate.

[59] This setting ensures the button can be copy/pasted into any slide without needing to adjust its timing.

[60] For example, the slide in which users make this choice might be available at any time through the slide Menu.

In-slide transcripts

If you're providing slide transcripts within a Storyline eCourse, it's recommended you not use Storyline's built-in Notes sidebar. Instead, present transcripts in-slide, such as through layers that:

A) Have the "Prevent the user from clicking on the other layers" slide layer property assigned to them

B) Include a "Transcript" heading (of the appropriate heading level) as a distinct layer element — i.e., not in the same text box as the transcript text — that is the highest transcript layer element in the slide's focus order

C) Obscure view of the slide content, through the transcript itself and, possibly, through incorporation of something like a semi-opaque shape

(A) helps ensure that screen reader focus returns to a predictable element in the slide's Focus Order when the layer is closed.

(A) and (B) combine to ensure that screen reader focus arrives on the correct element when the layer is opened and that the first element is automatically read, in effect providing an alert that the layer has opened.

(A) and (C) combine to ensure that all users have an equivalent experience when the transcript layer is open: screen reader users will be unable to read content on the base layer, or in other slide layers, due to the "Prevent the user from clicking on the other layers" property, while the transcript and, possibly, semi-opaque shape will make it so non-screen reader users are similarly unable to read content in the base layer or in other slide layers.

In-slide slide Menu

Using an in-slide slide Menu requires more work to produce and maintain than using Storyline's built-in Topbar slide Menu. It's recommended you not utilize the in-slide slide Menu approach unless you are willing to adopt this greater production and maintenance burden.

In-slide slide Menus will likely need conditions, variables, triggers, multiple button states and possibly even multiple button instances[61] to achieve some of the navigation restrictions made possible by the built-in Menu options.

Here are two sample ways you could program an in-slide Menu and all the related aspects within a course; the programming described in this section is shared between both samples linked below; later sub-sections will describe the programming differences between these two samples:

- Free navigation — i.e., no navigation restrictions
    - In-slide Menu Example (free navigation) demonstration
    - In-slide Menu Example (free navigation).story
- "Restricted" navigation[62]
    - In-slide Menu Example (restricted navigation) demonstration
    - In-slide Menu Example (restricted navigation).story

The in-slide Menus in these examples are facilitated through Storyline layers — hereafter referred to as "Menu layers" — that are opened/shown through a Menu button in each slide's base layer.

The Menu layers have the "Prevent the user from clicking on the other layers" slide layer property assigned to them. This ensures focus goes to the layer element highest in a slide's Focus Order and returns to a predictable spot in the slide's Focus Order when the layer is hidden.

---

[61] I.e., if you want slide Menu buttons to communicate which slides have been visited and which have not, *and* you also want to assign Down and/or Hover states to those buttons, you will likely need two instances of each button: one instance that communicates the slide has not been visited and a second instance that communicates the slide has been visited.

[62] I.e., the Menu can only be used to visit slides you have already visited through use of the eCourse's Next buttons

The Menu layers also have the “Pause timeline of the base layer” layer property assigned to them so base layer media does not play while the Menu is open.

The Menu layers also have their “Allow seeking” layer property set to “Yes” and their timelines set to as short a duration as possible. These two settings combine to ensure the seekbar isn’t misinterpreted or misused will the Menu layer is open. If “Allow seeking” is set to “No,” the seekbar will remain tied to the slide base layer while the Menu layer is open, meaning any user attempt to scrub the seekbar will actually impact the base layer, which users may not realize[63]. If “Allow seeking” is set to “Yes” or “Automatically decide” (which will typically decide “Yes”) *and* the layer timeline is not set to a short duration, the seekbar will appear to progress while the Menu layer is open, but that progression won’t actually be tied to anything.

The final, important layer property is setting “When revisiting” to “Reset to initial state”; this setting is safer than “Automatically decide” and helps ensure that any triggers tied to “when timeline starts” events are executed properly every time a user opens a Menu layer.

The Menu layer element highest in a slide’s Focus Order is text reading as “Menu” and programmed as a Heading 1.

Having the highest element be a short text string ensures screen reader users do not encounter the bug described in Addressing focus not landing at start.

Heading 1 would be the appropriate heading level for this heading since: A) the “Prevent the user from click on the other layers” layer property hides, while the layer is open, any heading hierarchy established in the slide’s base layer; and B) neither example includes the course title in the Course Player, which would automatically be programmed as a Heading 1 if present.

There are actually two Close buttons present in these Menu layers:

[63] E.g., consider this scenario: the base layer seekbar is 25% complete when a user opens the Menu layer; since the Menu layer is set to pause the base layer while open, the base layer is paused at the 25% complete mark, and the user can reasonably expect the base layer to resume form the 25% complete mark when they close the Menu layer. Except, let’s say that the user then scrubs the seekbar to 75% complete while the Menu layer is still open; this means that when the user closes the Menu layer, the base layer will resume from the 75% complete mark, and the user may not realize that their attempt to scrub the seekbar will the Menu layer was open caused them to miss 50% of the base layer.

One Close button is seen visually, includes a Down state and occurs early in the Focus Order, immediately after the Menu Heading 1 and before the slide buttons. This is the only close button mouse users will use. For assistive technology users, this particular close button provides a means to quickly close the Menu immediately after opening it[64].

The second Close button is hidden behind the first Close button, so it can only be encountered and used by assistive technology users (and because of that, it does not have a Down state). This button is last in the Focus Order and provides a way for assistive technology users to easily close the Menu after they've tabbed or read through it, thus sparing them from having to tab or read backwards through the Menu to reach the first Close button.

The Menu layers also include a "Hide this layer when user presses Ctrl+Alt+X after clicking on this layer" trigger, which provides another way for all users to quickly close the Menu. 'X' was chosen for this key press combination since it is not used by one of Storyline's built-in keyboard shortcuts and is commonly associated with close or exit actions.

A corresponding "Show Menu layer when user presses Ctrl+Alt+A after clicking on this slide" trigger is included in each slide's base layer. This trigger provides another way for all users to open a Menu layer. 'A' was chosen for this key press combination since it is not used by one of Storyline's built-in keyboard shortcuts, is easy to press in combination with Ctrl and Alt and invites learners to remember it via an "A for All slides" mnemonic device.

The Menu layer contains buttons that can be used to jump to other slides in the course. There are actually two instances of each slide's button: an "unvisited" instance, for when the slide has not yet been visited, and a "visited" instance that communicates the slide has been visited.

Whether or not the buttons communicate if a slide has been visited or not is left to developer discretion, though it is highly recommended that Menu buttons communicate this status, the same as Storyline's built-in menu does. If this status is communicated visually, it must also be communicated via button alt text, so that all users may access that information.

The demonstrations use two instances of each slide's button to communicate the unvisited/visited status because the buttons in these demonstrations have been given Down states (using the built-in Down state functionality), to help visually communicate they are buttons. If you did not use Down states, you could use a single instance of each slide's button to communicate the unvisited/visited status.

[64] As a user might want to do if they opened the Menu by mistake.

Conditional triggers associated with the slide buttons control which instance displays at any given time. There is a true/false Project variable that tracks if each slide has been visited. Each of these variables starts with the default value False, and each slide contains an “adjust variable when timeline of this slide starts” trigger that will toggle the corresponding visit variable to True. If a slide’s visit variable is False, the “unvisited” button instance will remain in its normal state, and the “visited” button instance will have its state changed to hidden. If a slide’s visit variable has been toggled to True, the “unvisited” button instance will have its state changed to hidden, and the “visited” button instance will remain in its normal state.

The slide button instances corresponding with the slide a user is currently visiting are treated differently. The “unvisited” instance has been deleted. The “visited” instance is given a distinct visual appearance and distinct alt text: “Currently visiting *slide title*.” The “visited” instance’s triggers and Down state have been deleted, and its Normal state, with distinct appearance and alt text, has been duplicated as the built-in Disabled state. Finally, the Disabled state is assigned as the button’s default state. All these settings combine to create the effect of, when a screen reader user is reading through the Menu’s slide buttons and they encounter the button associated with the slide they’re currently on, it reads to them as “Disabled currently visiting *slide title*” and cannot be used (since they’re already on that slide); keyboard navigators will automatically tab past this button.

The slide buttons have all been placed in a scrolling panel, since there are more buttons than would otherwise fit. Alternatively, the buttons could be arranged in multiple columns so that a scrolling panel was not necessary.

A semi-opaque shape, hidden from accessibility tools, is placed behind the Menu, obscuring learner’s ability to view base layer content while the Menu layer is opened. This shape is necessary so that non-screen reader users have an equivalent experience as screen reader users: that is, screen reader users are unable to read base layer content when the Menu layer is open due to that layer having been assigned the “Prevent the user from clicking on the other layers” layer property, so other users must be similarly unable to read base layer content.

Free navigation version

In this version, the “unvisited” button instances do not have an icon that communicates their status, and they can be used at any time.

Restricted navigation version

In this version, the “unvisited” button instances have several distinct properties:

1. The have a lock icon that communicates visually that they cannot be used
2. They do not have a “Jump to slide…” trigger, as they cannot be used (because with “restricted navigation,” you cannot use the Menu to visit a previously unvisited slide)
3. They do not have a Down state (because they cannot be used)
4. Instead of having a Down state, their Normal state has been duplicated as the built-in Disabled state, and the Disabled state has been assigned as the default state (this helps communicate to screen reader users that the buttons cannot be used)