

Wiki Accessibility & Design

iGEM uniCRETE

Conducted By Stavros Piperakis

20/10/2026

Controls that adapt

Designing a wiki that adapts around user preferences is an essential step to providing information in a way the users may already feel familiar. However providing too many options also creates friction especially to users with minimal technical familiarity. For this reason it is critical to implement components that form around preexisting user choices, such as preferred device or assistive technologies.

Our approach was to create a modular system with parameters that could act as building blocks of the design that could co-exist and work together. The goal was to create a design whose building components did not interfere with each other and could form together a more optimal experience and target a broader set of users.

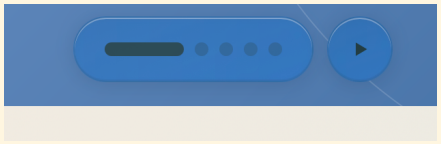
The main pillars implemented in our wiki for adaptivity are the following:

- Dynamic Type - Adapting typography and text sizing to improve readability without disrupting the structure of the page.
- Transparency Reduction - Reducing transparency and visual layering where these effects may make content harder to distinguish.
- Motion Reduction - Minimizing animations and transitions for users who prefer reduced motion.
- High Contrast - Increasing the visual distinction between foreground and background elements to improve readability and navigation.
- Color Vision Filters - Providing alternative color presentations to make information distinguishable for users with different forms of color vision deficiency.
- Dark Mode - Providing a darker visual environment for users who prefer reduced luminance or dark interfaces.

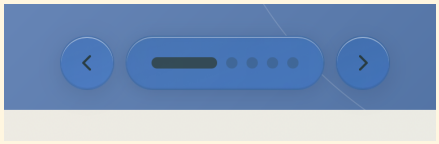
Each of those require deeper consideration on how to optimize interaction in specific options. Sometimes interaction models deeply rely on motion or transparency in which case the experiences need to be reconsidered and adapted.

Explore more section, available to all of our wiki pages is one of those interactions. Reduce motion would work without any changes to controls, however, this would limit users ability to move in an organic way, as autoplay is disabled in this setting.

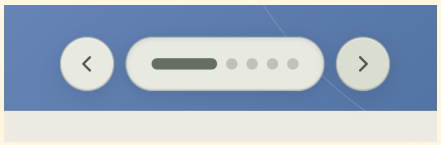
Here are a few examples from our wiki Explore More section to demonstrate this:



Default autoplay controls



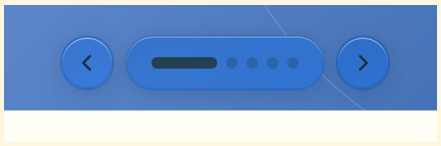
Reduced motion controls



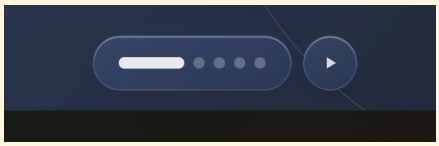
Reduced motion | Reduced transparency



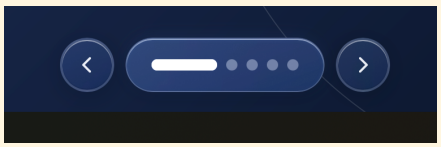
Reduced transparency



Reduced motion | High contrast



Default dark mode controls



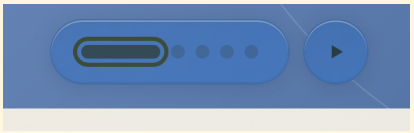
Dark mode | Reduced motion



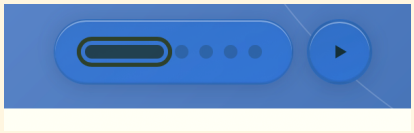
Dark mode | Reduced motion | Reduced transparency

High Contrast Mode

High contrast is often treated as a color adjustment, but color alone is rarely enough. A visitor who needs stronger separation between foreground and background is frequently the same visitor for whom color is an unreliable signal in the first place. For this reason our high contrast mode does not only push contrast and saturation: it adds non-color signals wherever color was doing the work on its own. Links become permanently underlined rather than merely colored, interactive borders double in width, and the focus ring grows from two pixels to three. A visitor navigating by keyboard in high contrast gets a stronger indicator precisely because they are the visitor most likely to depend on it.

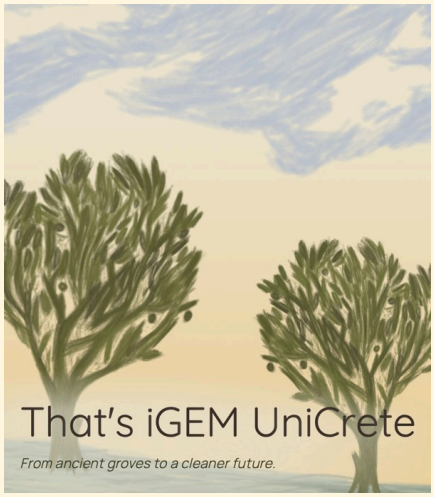


Default Keyboard Navigation Ring

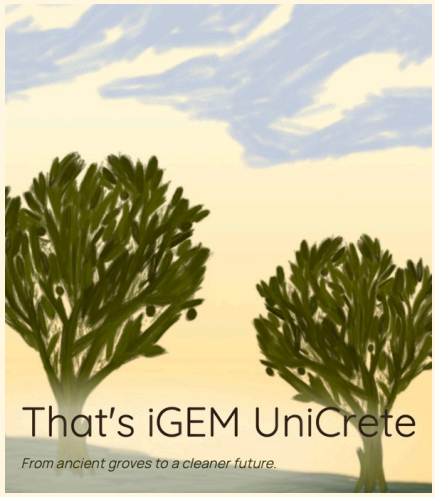


High Contrast Navigation Ring

Another example, in our Hero Page, under high contrast the tree silhouettes reach 3.39:1 against the sky and the ground 1.64:1, and the adjustments were tuned until the horizon edge stayed readable in every combination.



Default Hero Page



High Contrast Hero Page



Default Hero Page, dark mode



High Contrast Hero Page, dark mode

Color Vision Filters

About 8% of men and 0.5% of women of northern European descent have red-green color vision deficiency [1]. Our Engineering page uses four colors to mark the four phases of a DBTL cycle, so this was never a styling question: if those colors collapse into each other, the page stops explaining itself.



The engineering cycle widget has been designed for minimal change across all color-blindness modes

The mode does two things, kept as two switches. It rebuilds the colors that carry meaning, pulling apart pairs that would otherwise look the same, and it passes images through a daltonization filter [2] that moves information out of the color channel the viewer cannot see into ones they can. They are separate because some visitors want corrected diagrams but find recolored photographs strange, and that is their call.

One mode deliberately does less. Tritanopia affects blue and yellow but leaves red and green intact [3], so plates, gels and fluorescence already read correctly. We tried filtering them anyway and measured thirteen color pairs from the site: six of them got worse in perceptual distance. Tritanopia mode now adjusts the palette and leaves images alone, with the reason written into the stylesheet so it is not “fixed” later and users are informed in the panel.

Touch based interaction

Achieving operability in touch-based devices is also fundamental, hover based controls as well as interactions that require mouse level accuracy have to be optimized. This requires keeping a minimal and approachable design on both interfaces that keeps all the necessary information for the user to understand how to interact with each experience.

On a mouse-controlled environment users have the option to hover to reveal more information about the action they are about to take, this is not an option in touch. For this reason, on touch-based interfaces all controls need to be more approachable and direct.

Why working with existing assistive technology matters more than any panel

User Defaults and Interoperability

Most people with specific preferences will probably have already made a choice, saved on their device. Preferences as Reduced Motion, Reduced Transparency or High Contrast are usually set in Linux based systems, macOS, Windows, or iOS earlier, and it follows users everywhere.

Most of the times, reinventing the wheel produces adverse effects and creates barriers that have no reason to exist. Users of assistive technologies have personal experiences that helped them set up exactly to their preferences. A person using a screen reader has spent time configuring its verbosity, its voice, its rotor. An accessible wiki should respect these choices rather than asking users to configure them again.

So, the panel starts from what the browser already reports [4]. Where a supported preference is available, it is inherited on first visit, without anyone opening anything:

What The Visitor Already Set	What The Wiki Reads	Applied On First Load By Default
Dark Appearance	prefers-color-scheme: dark	Dark Theme
Reduce Motion	prefers-reduced-motion: reduce	Animations Off, Requiring Interaction Redesign
Increase Contrast	prefers-contrast: more	High Contrast Mode
Reduce Transparency	prefers-reduced-transparency	Blur and Translucency Removed, Requiring Interaction Redesign

An explicit choice in our panel is written to LocalStorage and always has a priority compared to defaults. This allows users to experiment, only if they feel comfortable to, always respecting their choices and boundaries.

The other part: Compatibility with User Personalization Software

Inheriting preferences is one part of the picture. Ensuring compatibility with software that visitors bring with them is another, and perhaps the more demanding one.

We cannot see these tools, but we can still break them.

Testing with the most common ones is required. Following Web Content Accessibility Guidelines (WCAG) [5] usually results in a smooth operability with most screen readers and other software.

Existing Technology	What Would Break It	What We Do Instead
Screen readers (VoiceOver, NVDA, JAWS)	Decorative role on links, aria-hidden over focusable content, empty alt on meaningful images	Links keep their real role, focusable content is never hidden, and meaningful images always get descriptive alt text
Windows High Contrast	forced-color-adjust: none; focus rings drawn with filter or shadow, which the mode discards	A dedicated stylesheet restores a real outline on all eight controls that suppress it

Only the most needed information must be shared

An important privacy consideration. Assistive technologies generally do not need to identify themselves to web content [6]. This is intentional: exposing such information could enable users of assistive technologies to be distinguished or fingerprinted. A great example would be eye tracking [7], which could be weaponized against user intent if it would provide more information than needed for interaction.

Screen Readers

A screen reader turns a page into speech or braille, one element at a time. What matters most is the structure beneath the visual design to provide an organized and easy to navigate experience.

We started with the basics, semantic HTML, real headings, lists, links and buttons, and a standardized page language to allow users to browse the page, within its contents as well as in other areas of the page.



Skipping the menu with one key press

Images that carry information have text alternatives. On our Home page, each clay scene that illustrates the problem is a single image with a description of what it shows, not a stack of layers read out one by one. Hiding what is purely visual matters just as much. In the DBTL wheel, internal graphic definitions were being listed as unnamed

images. On the Entrepreneurship page, the axis labels of the risk matrix were read as a stream of disconnected words. Both are now hidden, and the risk ratings remain available in the list below the chart. Decorative symbols are hidden as well, such as the star on a "Recommended" card, which a screen reader would otherwise announce as "black star".

Changes that happen without a page reload are announced. This includes gallery counters as well as changes in the accessibility panel and of course included experiences like our Bacto Builder game.

Our education game, Bacto Builder, takes accessibility a step further, it can be played by sight, by ear with left and right sound and spoken cues, by keyboard, or touch. It has detailed descriptions of the bacteria children imagined during their visit at our booth at the Researcher's Night at University of Crete. We are truly proud of this experience and can't wait for you to try it.

Our experience has mainly been tested using VoiceOver, the most widely used screen reader on mobile devices [8]. Screen readers rely on the browser to expose the page as an accessibility tree through each platform's accessibility API, following the W3C Core Accessibility API Mappings [9], so correct semantics generally carry over to other screen readers as well. However, we understand getting feedback from the same communities that will get to experience those features is of the highest importance.

Getting feedback from the community

This is why we met Konstantinos, who is blind and navigates the web with a screen reader. We spoke twice over the course of the wiki's development. The first call was in the early stages, to get to know each other and see whether our early navigation elements were operable with screen readers. The second was more thorough and gave us greater insight into how to finalize our structure, as we tested two of the most important pages of our wiki: the Home page and the Entrepreneurship page.

The Home page had several points of improvement. The scroll-locked sections caused several elements to be read more than once, and the images that together form each landscape were read as separate pieces rather than as one scene. We had to merge them into a single image with a single description, which is how the clay scenes

described above came to be. Konstantinos also gave us a simple rule for writing those descriptions: every alt text has to say what is in an image and where it is, so a listener can picture the scene the way a sighted visitor sees it. The Entrepreneurship page, on the other hand, had a much better structure from the first attempt.

We also got in touch a bit before the wiki freeze to get our final feedback, and he verified that there was an observable improvement in the overall accessibility of the page after our restructure.

This does not mean that our solution is perfect. Based on our discussions not supporting more languages other than English could be limiting especially considering that some screen readers read each page with the voice and pronunciation to the language it declares. A visitor whose first language is Greek hears the entire wiki in English, and following a second language by ear, one element at a time, is considerably harder than reading it. A translated version, with each language correctly declared so screen readers can switch voices on their own, would open the wiki to the local communities around us, and it is the first improvement we would recommend for the next team.

Keyboard only navigation

Most platforms have been designed around the use of multiple methods of input typically mouse and keyboard or touch. There are a few instances in which users might find it easier to navigate only using keyboard, including people with motor impairments, users who have difficulty using a mouse, and those who rely on assistive technologies.



Focus ring elegantly shows where the next interaction is

It is of high importance to ensure all interactive elements and information can be accessed and operated using a keyboard alone.

The first Tab on any page reveals a "Skip to main content" link, so visitors do not have to pass through the whole navigation bar on every page. From there, focus follows the reading order of the page, and a focus ring always shows where it is.

Menus, carousels and galleries follow the W3C ARIA Authoring Practices [10], so they behave the way keyboard users already expect from desktop software. Navigation dropdowns open with Enter, Space or the down arrow. The arrow keys move between items, Home and End jump to the first and last item, and Esc closes the menu and returns focus to the button that opened it. Galleries and carousels move with the left and right arrows.

In Explore More, the slide controls form a single toolbar with one tab stop, so passing it takes one Tab instead of one for each card. When focus enters a carousel, autoplay pauses, so content never moves while someone is reading it.

Content that is hidden stays hidden from the keyboard too. While closed, the mobile menu, the accessibility panel and collapsed menu groups are inert, so Tab never lands on something the visitor cannot see. When the accessibility panel is open, focus stays inside it until Esc closes it, and then it returns to the button that opened the panel.

Accessibility features improve the experience for everyone

Personalizing the experience and making it more adaptive around the visitor of our wiki is great, but sometimes it is also important to design with the capabilities of each user’s device in mind.

We all like impressive details in the designs of modern web pages, but access to information must remain open even for those with devices with limited capabilities. Reduce Transparency and Reduce Motion can significantly improve performance on older devices and especially mobile devices.

Frosted glass is one of the most demanding effects a browser can draw, because the blur behind each panel must be redrawn whenever the content underneath moves. Reduce Transparency removes every blur on the wiki at once. Reduce Motion stops parallax, autoplay and animated transitions, so the page only changes when the visitor asks it

to. On older phones or laptops, this can make scrolling smoother and use less battery.

We use the same principle of adaptation, on devices with fewer than 4 cores or less than 4gb of memory, the illustrated scene on the Home Page stays still, exactly as it does under reduced motion.

When a visitor pinches to zoom on the Team Page, the phone must redraw every blur at the larger size. This can quickly result in enough memory usage to close the tab to maintain system stability. So, while the page is zoomed in, the blurs and filters step aside, the name labels become nearly solid so they stay readable. This also helps improve readability for users who need to pinch to zoom. Once the page returns to its normal size, the visual effects are restored.

In our testing, as described in each figure, we could achieve substantial improvements in performance that could make a difference for many users.

Test setup

Environment: Chrome 155 running headless on a Mac (MacBook Air M3 16/512gb), emulating a mobile phone: a 390×844 screen at 3× pixel density, touch input, and a CPU slowed down 4×, the same throttling Lighthouse applies for mobile.

Test: each page scrolls from top to bottom at a constant speed for 8 seconds. Each setting was measured three times, and we report the median.

Enabling the settings: through the same LocalStorage keys that the accessibility panel writes.

Page	Render Passes Per Frame	Drawing Time Per Frame
Team	7.3 → 0.5 (−93%)	0.53 → 0.04 ms (−92%)
Description	3.5 → 0.35 (−90%)	0.25 → 0.05 ms (−80%)
Home	2.4 → 0.5 (−80%)	0.27 → 0.11 ms (−59%)

Reduce Transparency, measured in Chrome on an Apple M3 laptop. Each cell shows the default value and the value with Reduce Transparency on. Removing the blurs cuts the render passes per frame by 80 to 93% and the drawing time per frame by up to 92%, with the largest gain on the Team page.

Page	Default	Reduce Motion	Both Settings
Home	1096 ms	266 ms (−76%)	294 ms (−73%)
Team	614 ms	283 ms (−54%)	152 ms (−75%)
Description	507 ms	207 ms (−59%)	123 ms (−76%)

Main thread time while scrolling each page, on the same setup. Reduce Motion alone lowers it by 54 to 76%, and combined with Reduce Transparency the Team and Description pages drop by about three quarters.

Page	Default	Reduce Transparency
Team	7.0 ms	0.57 ms (−92%)
Description	5.5 ms	0.56 ms (−90%)

The worst case: the same setup with the GPU disabled, so every blur is drawn by the processor. A 60 Hz display leaves 16.7 ms for each frame, and on the Team page the blurs alone take 7 ms, about 42% of it. Reduce Transparency brings this down to 0.57 ms.

References

[1] A. Almustanyir, “A Global Perspective of Color Vision Deficiency: Awareness, Diagnosis, and Lived Experiences,” Healthcare, vol. 13, no. 16, Art. no. 2031, 2025, doi: 10.3390/healthcare13162031.

[2] G. M. Machado, M. M. Oliveira, and L. A. F. Fernandes, “A Physiologically-based Model for Simulation of Color Vision Deficiency,” IEEE Transactions on Visualization and Computer Graphics, vol. 15, no. 6, pp. 1291-1298, 2009, doi: 10.1109/TVCG.2009.113.

[3] National Eye Institute, “Types of Color Vision Deficiency,” National Institutes of Health. [Online]. Available: https://www.nei.nih.gov/learn-about-eye-health/eye-conditions-and-diseases/color-blindness/types-color-vision-deficiency (accessed Sep. 13, 2026).

[4] W3C CSS Working Group, “Media Queries Level 5,” W3C Working Draft, Feb. 2026. [Online]. Available: https://www.w3.org/TR/mediaqueries-5/ (accessed Sep. 12, 2026).

[5] W3C, “Web Content Accessibility Guidelines (WCAG) 2.2,” W3C Recommendation, Oct. 2023. [Online]. Available: https://www.w3.org/TR/WCAG22/ (accessed Sep. 21, 2026).

[6] W3C Technical Architecture Group, “Web Platform Design Principles, §2.11: Don’t reveal that assistive technologies are being used,” W3C Group Note, Sep. 2026. [Online]. Available: https://www.w3.org/TR/design-principles/#do-not-expose-use-of-assistive-tech (accessed Sep. 21, 2026).

[7] J. L. Kröger, O. H.-M. Lutz, and F. Müller, “What Does Your Gaze Reveal About You? On the Privacy Implications of Eye Tracking,” in Privacy and Identity Management. Data for Better Living: AI and Privacy, IFIP AICT, vol. 576. Cham, Switzerland: Springer, 2020, doi: 10.1007/978-3-030-42504-3_15.

[8] WebAIM, “Screen Reader User Survey #10 Results,” Feb. 2024. [Online]. Available: https://webaim.org/projects/screenreadersurvey10/ (accessed Oct. 20, 2026).

[9] W3C, “Core Accessibility API Mappings 1.1,” W3C Recommendation, Dec. 2017. [Online]. Available: https://www.w3.org/TR/core-aam/ (accessed Oct. 4, 2026).

[10] W3C Web Accessibility Initiative, “ARIA Authoring Practices Guide (APG): Patterns,” [Online]. Available: https://www.w3.org/WAI/ARIA/apg/patterns/ (accessed Oct. 3, 2026).