VIS2026

Workshop on Visualization for Accessibility · Boston

Visual Accessibility Auditing Across Interaction States and Reading Orders

Site Accessibility Auditor: a Chrome DevTools extension

Rahat Zaman · Jake Wagoner · Tingying He · Alexander Lex · Paul Rosen

SCI Institute · University of Utah · TU Graz

Web accessibility is now a legal obligation for public websites

Accessibility on the Web

Web accessibility is now a legal obligation for public websites

Automated checkers are indispensable

Missing labels · invalid ARIA · low color contrast



Lighthouse

deque

axe-core

WAVE
web accessibility evaluation tool

WAVE

Accessibility on the Web

Web accessibility is now a legal obligation for public websites

Automated checkers are indispensable

Missing labels · invalid ARIA · low color contrast



Lighthouse

deque

axe-core

WAVE
web accessibility evaluation tool

WAVE

56 average WCAG 2 failures on each of top million home pages

Accessibility on the Web

Web accessibility is now a legal obligation for public websites

Automated checkers are indispensable

Missing labels · invalid ARIA · low color contrast

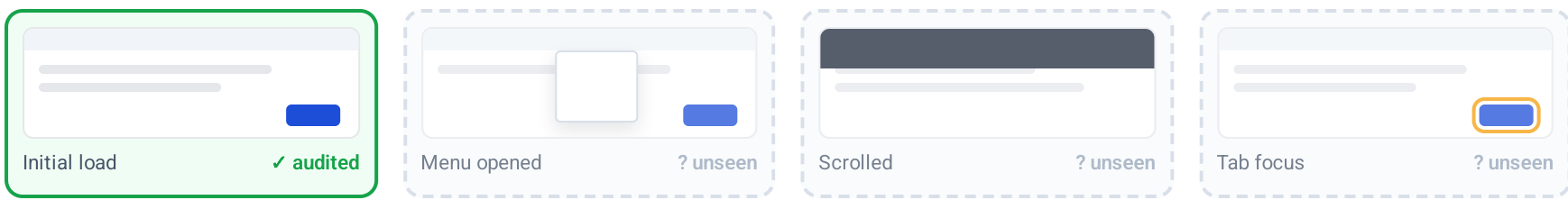
Lighthouse

deque
axe-core

WAVE
web accessibility evaluation tool
WAVE

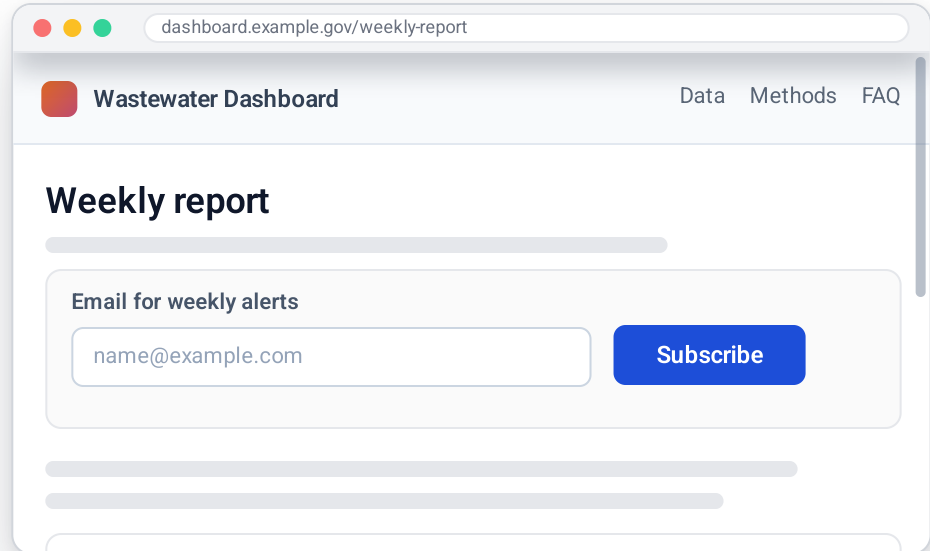
56 average WCAG 2 failures on each of top million home pages

Each checker evaluates a single snapshot of the DOM, but users reach many more states.



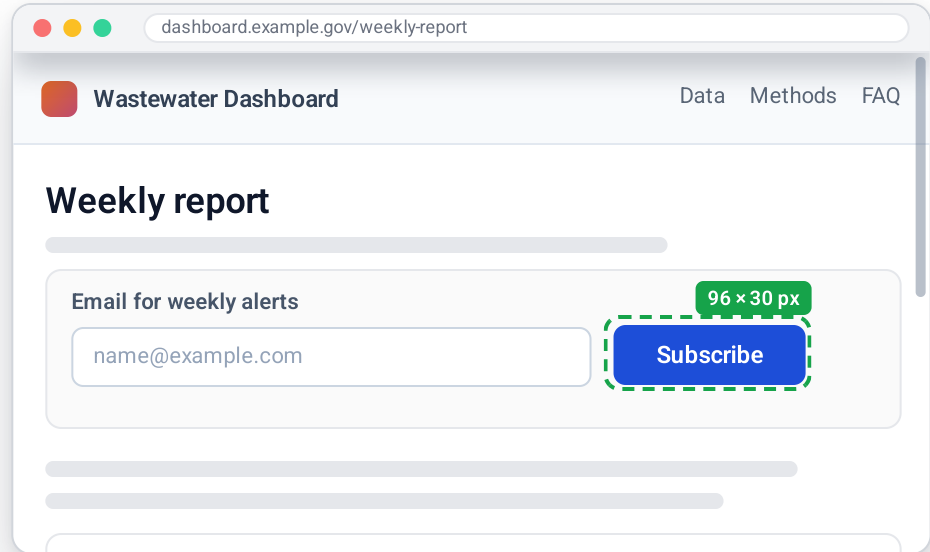
Problem 1: barriers that appear only after interaction

- `2.5.8` Target Size (Minimum)
- `2.4.7` Focus Visible
- `2.4.11` Focus Not Obscured
- `1.4.11` Non-text Contrast of focus
- `2.5.7` Dragging Movements
- `3.3.7` Redundant Entry
- `3.3.8` Accessible Authentication
- `3.2.6` Consistent Help



Problem 1: barriers that appear only after interaction

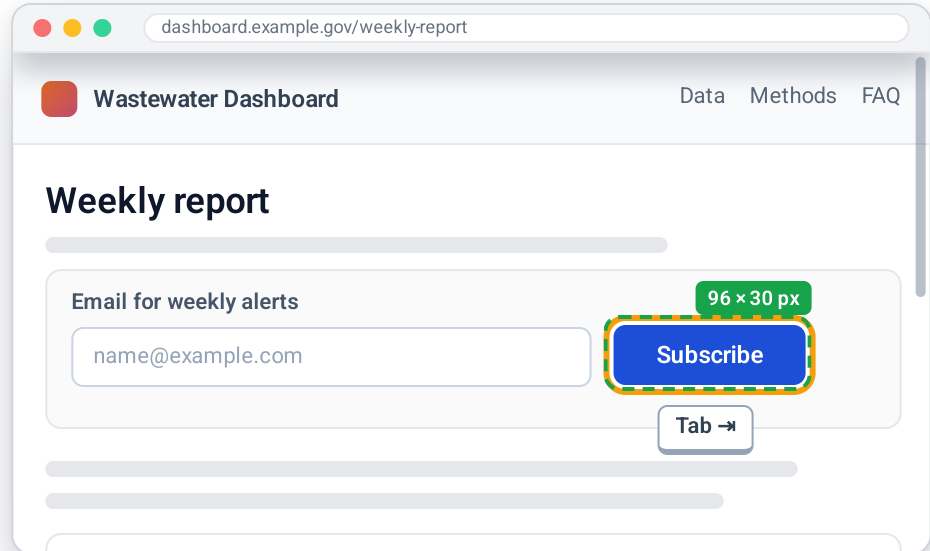
- `2.5.8` Target Size (Minimum) ✓
- `2.4.7` Focus Visible
- `2.4.11` Focus Not Obscured
- `1.4.11` Non-text Contrast of focus
- `2.5.7` Dragging Movements
- `3.3.7` Redundant Entry
- `3.3.8` Accessible Authentication
- `3.2.6` Consistent Help



2.5.8 Target size ✓

Problem 1: barriers that appear only after interaction

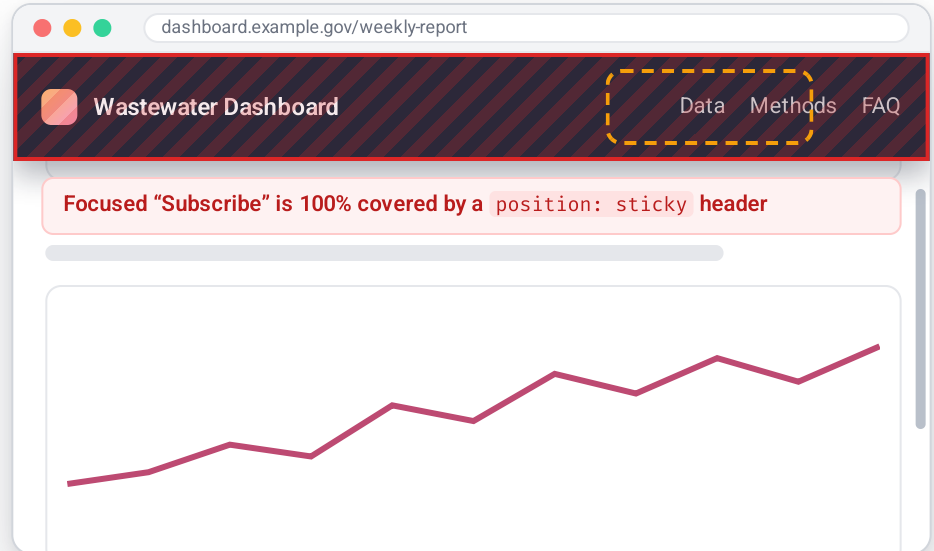
- `2.5.8` Target Size (Minimum) ✓
- `2.4.7` Focus Visible ✓
- `2.4.11` Focus Not Obscured
- `1.4.11` Non-text Contrast of focus
- `2.5.7` Dragging Movements
- `3.3.7` Redundant Entry
- `3.3.8` Accessible Authentication
- `3.2.6` Consistent Help



2.5.8 Target size ✓ **2.4.7** Focus visible ✓

Problem 1: barriers that appear only after interaction

- `2.5.8` Target Size (Minimum) ✓
- `2.4.7` Focus Visible ✓
- `2.4.11` Focus Not Obscured ✗
- `1.4.11` Non-text Contrast of focus
- `2.5.7` Dragging Movements
- `3.3.7` Redundant Entry
- `3.3.8` Accessible Authentication
- `3.2.6` Consistent Help



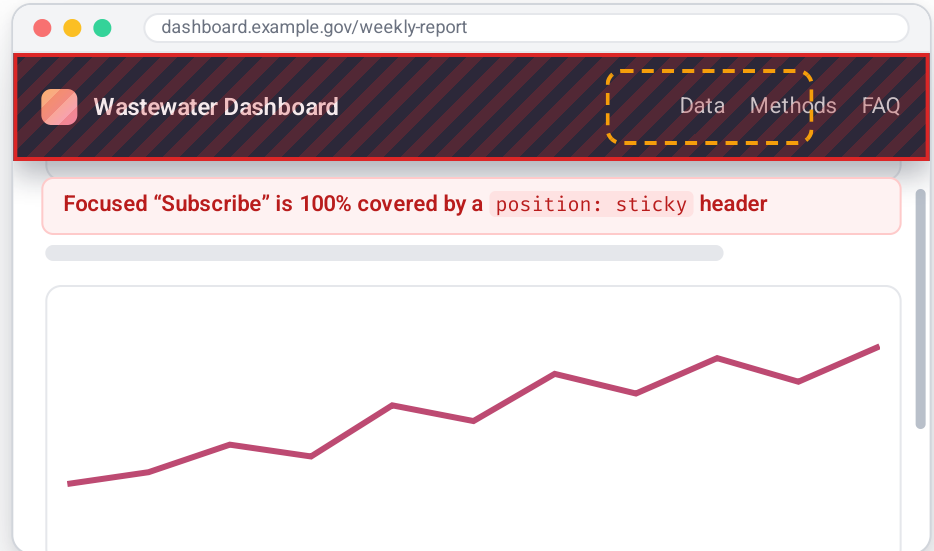
2.5.8 Target size ✓ 2.4.7 Focus visible ✓ 2.4.11 Focus not obscured ✗

Problem 1: barriers that appear only after interaction

- `2.5.8` Target Size (Minimum) ✓
- `2.4.7` Focus Visible ✓
- `2.4.11` Focus Not Obscured ✗
- `1.4.11` Non-text Contrast of focus
- `2.5.7` Dragging Movements

need memory across states

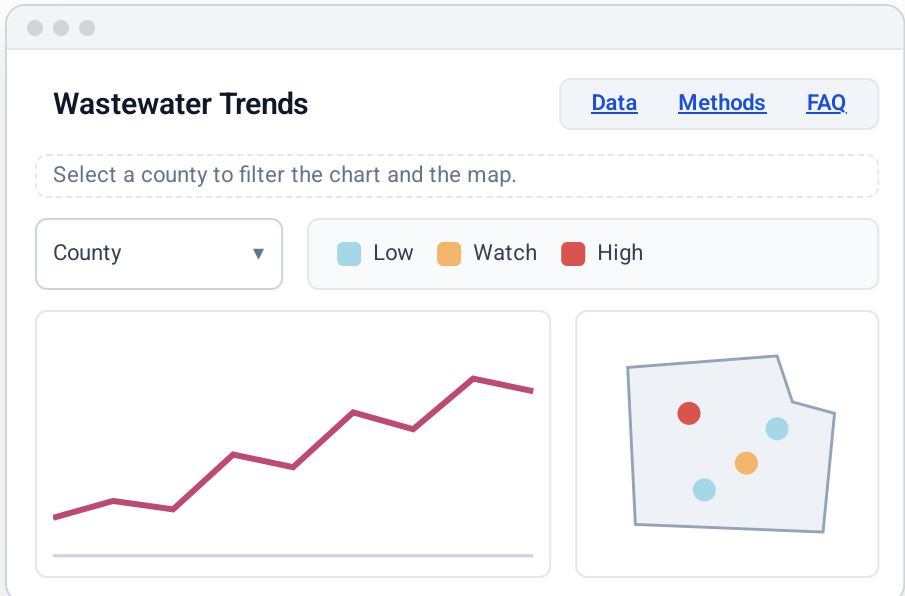
- `3.3.7` Redundant Entry
- `3.3.8` Accessible Authentication
- `3.2.6` Consistent Help



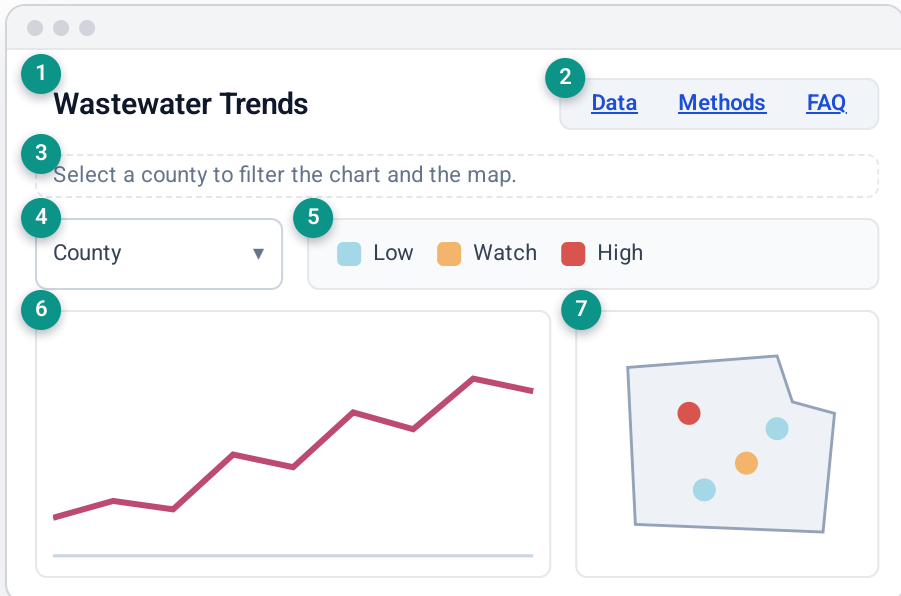
2.5.8 Target size ✓ | 2.4.7 Focus visible ✓ | 2.4.11 Focus not obscured ✗

Initial page load **PASS** → After scrolling and pressing Tab **FAIL**

Problem 2: visual, screen-reader, and keyboard orders *disagree*



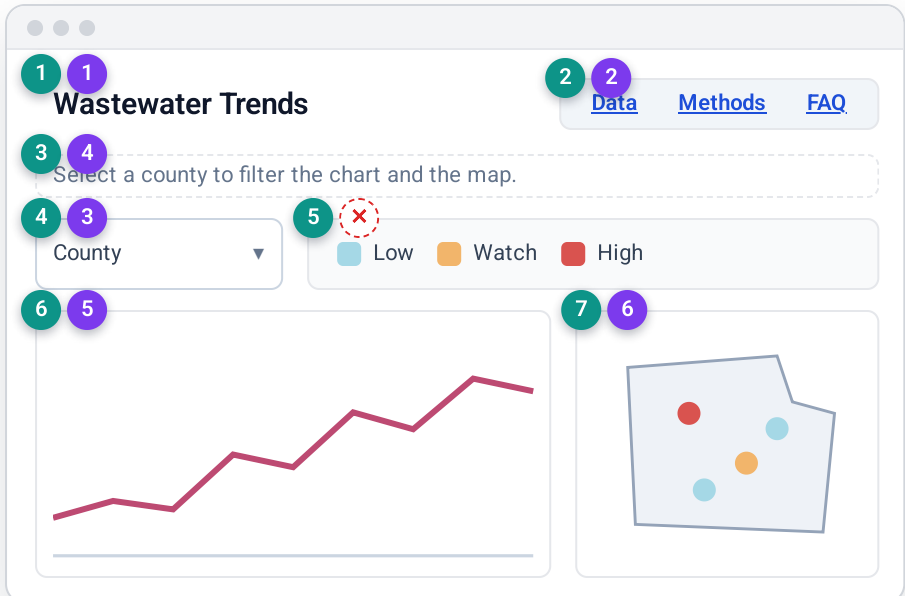
Problem 2: visual, screen-reader, and keyboard orders *disagree*



Visual order

Layout read row by row: what sighted readers follow.

Problem 2: visual, screen-reader, and keyboard orders *disagree*



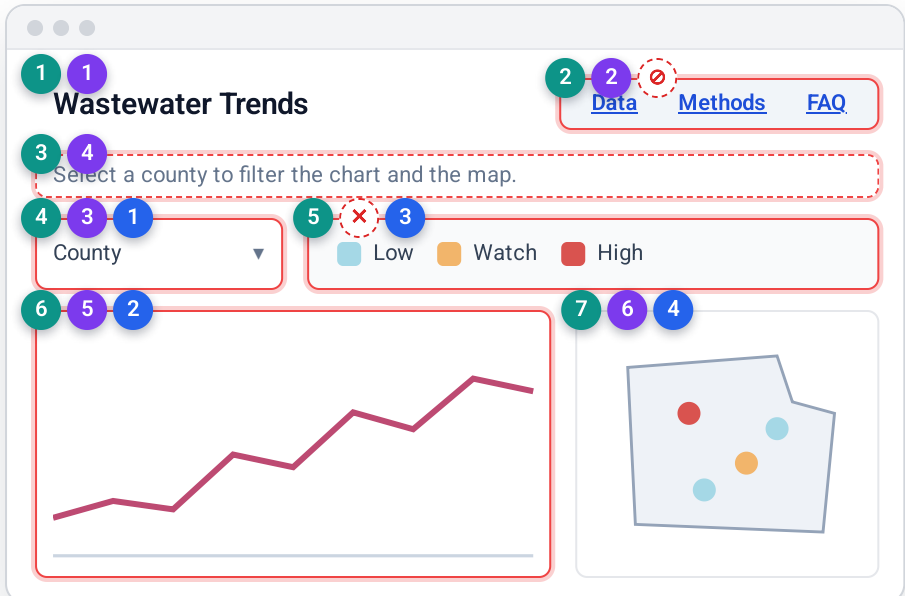
Visual order

Layout read row by row: what sighted readers follow.

Screen-reader order

Accessibility tree: the filter is announced **before its instructions**, and the legend is **missing**.

Problem 2: visual, screen-reader, and keyboard orders *disagree*



Visual order

Layout read row by row: what sighted readers follow.

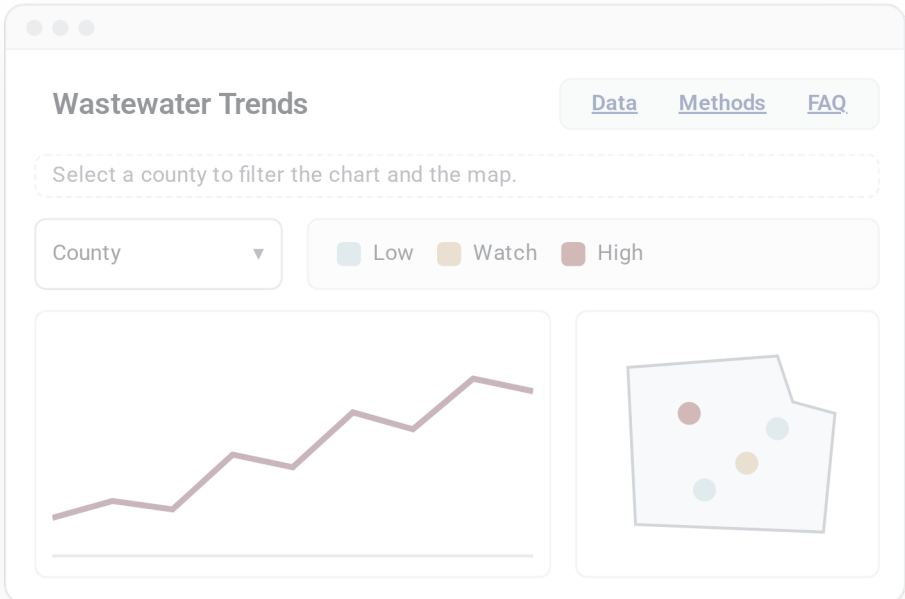
Screen-reader order

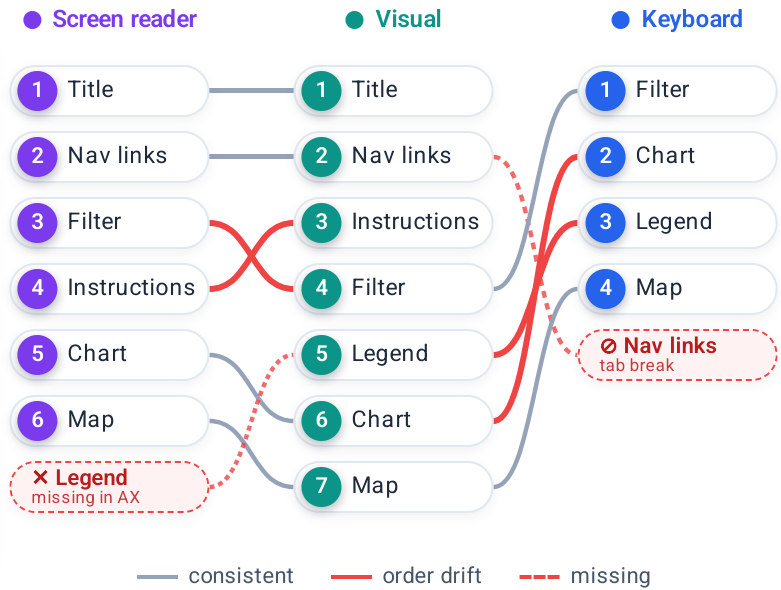
Accessibility tree: the filter is announced **before its instructions**, and the legend is **missing**.

Keyboard order

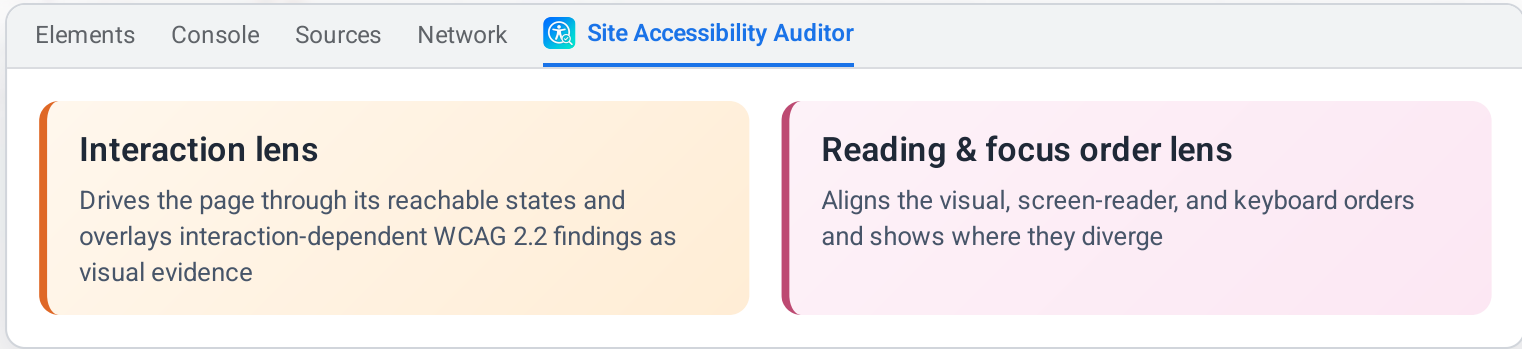
Tab sequence: the nav links are **skipped** (tab break), and the chart comes **before its legend**.

Problem 2: visual, screen-reader, and keyboard orders *disagree*



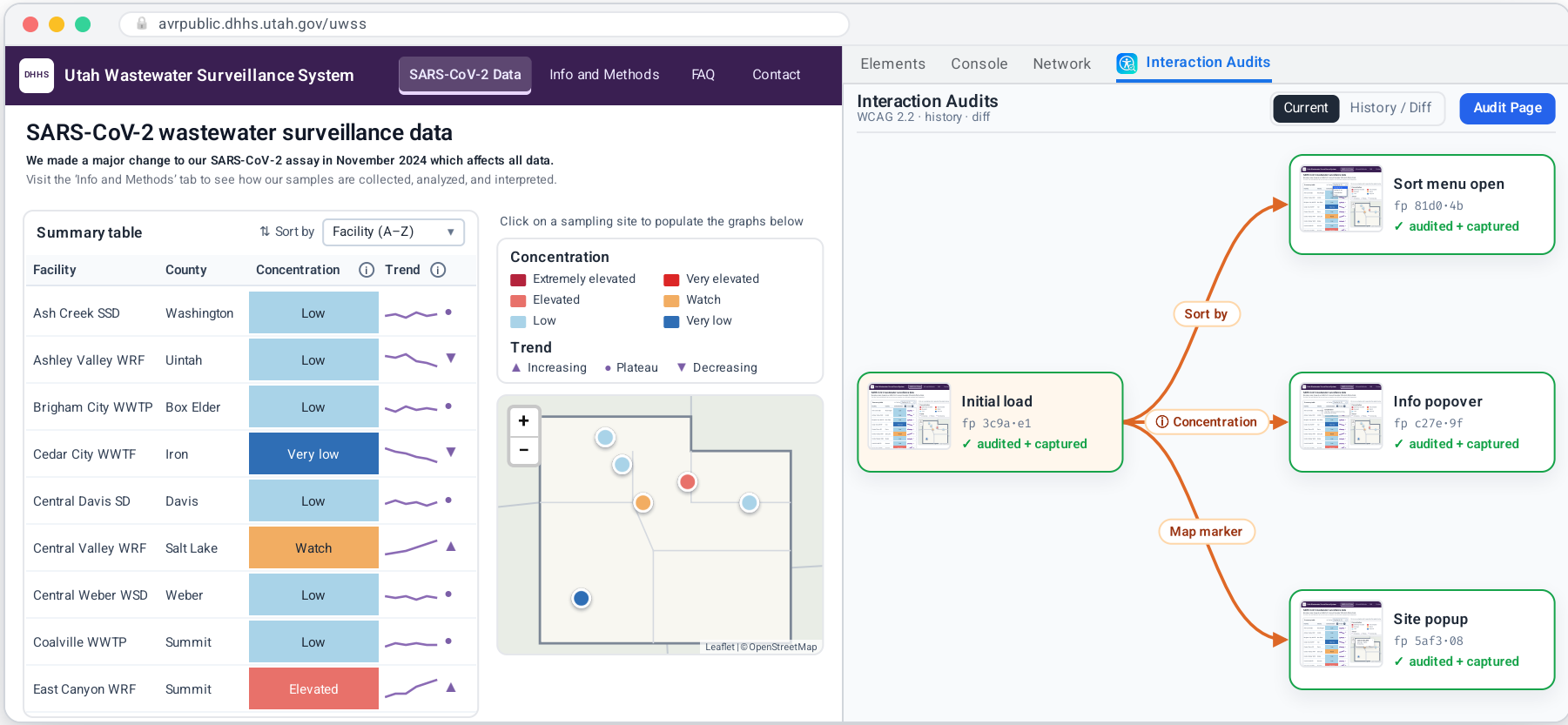


Treat accessibility auditing as a visualization problem: show interaction-dependent and order-dependent evidence on the live page, where repairs happen.



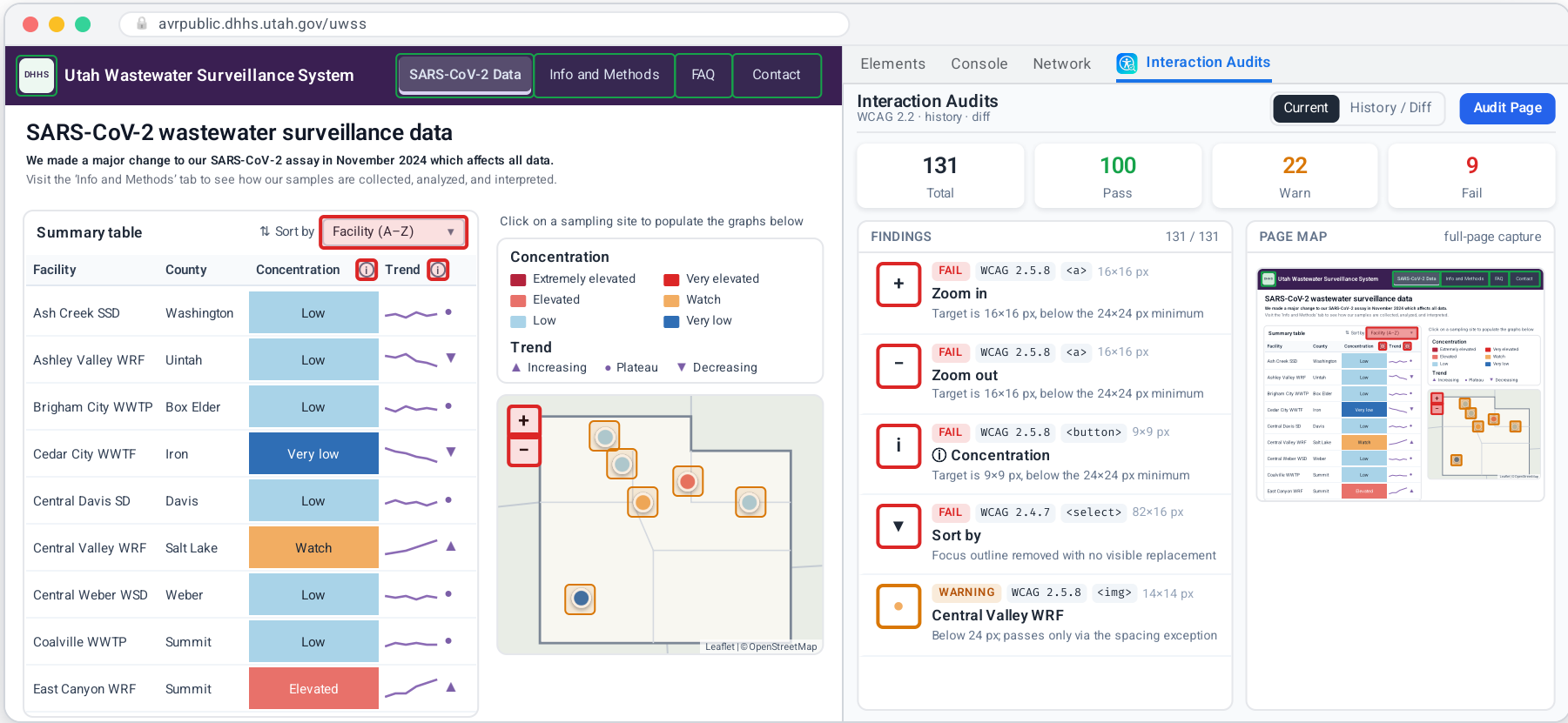
Interaction lens

audits every reachable state, not just the first one



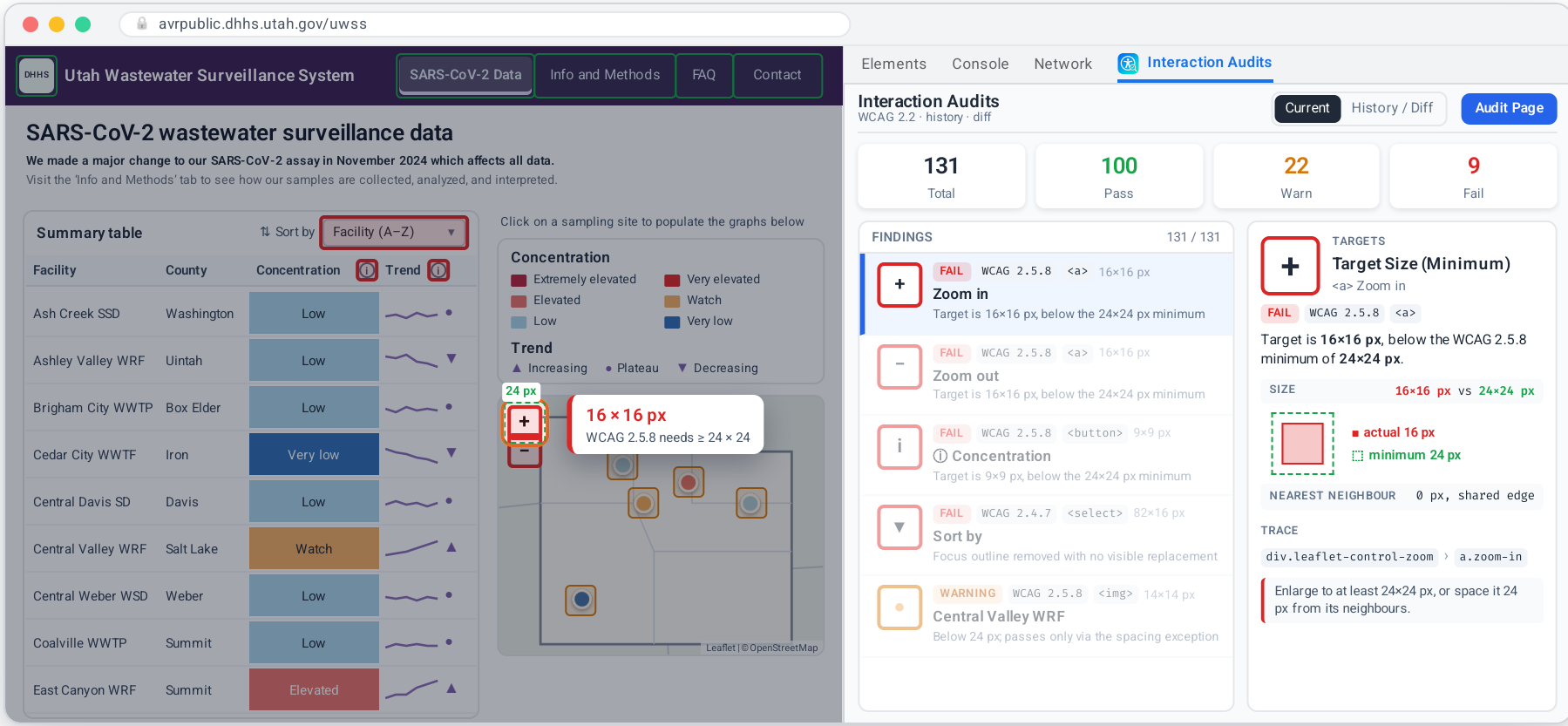
1 Drives the live page through its **reachable states** and audits every one, not just the first

Interaction lens audits every reachable state, not just the first one



2 Findings become **visual evidence** on the page: **pass**, **warning**, **fail** for 131 targets

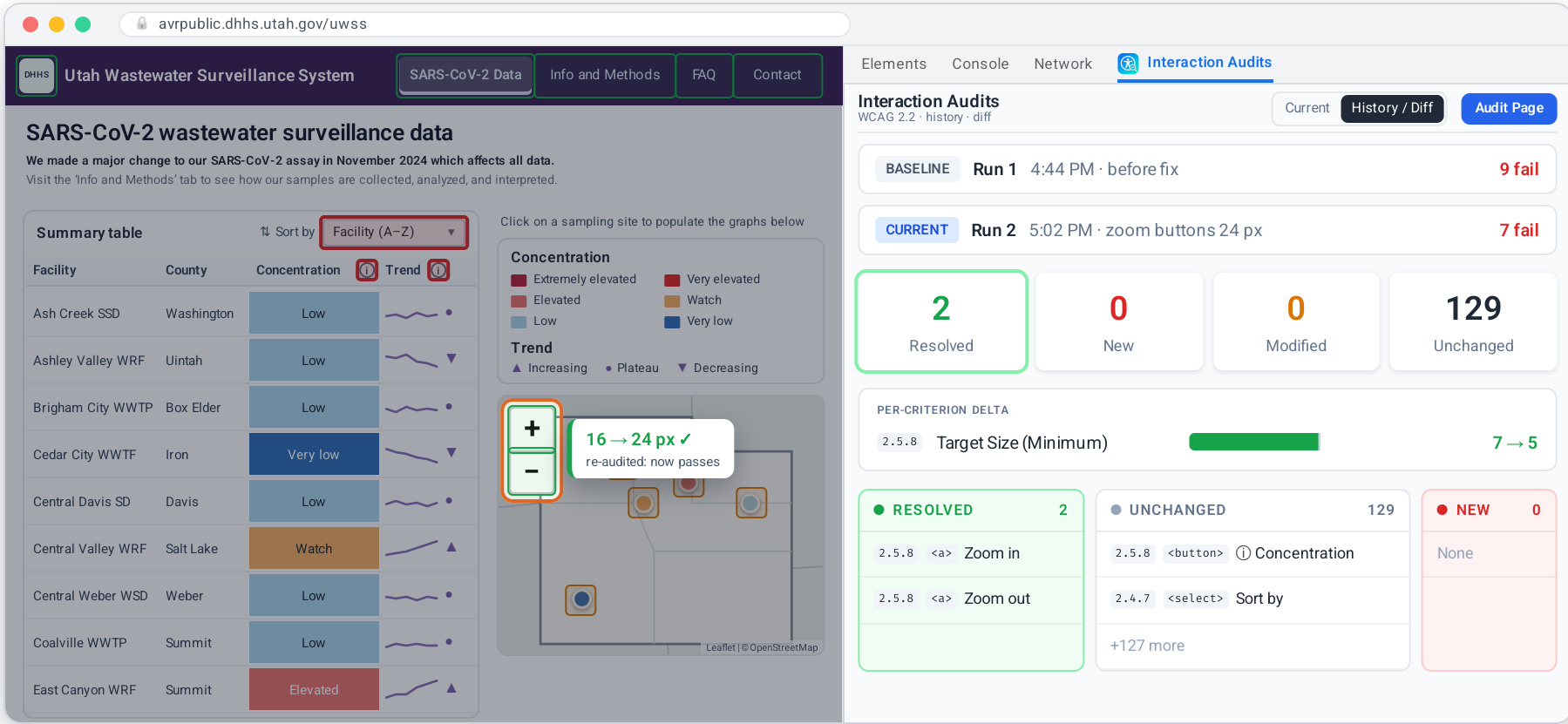
Interaction lens audits every reachable state, not just the first one



3 Each finding is **inspectable**: why it failed, where it is in the DOM, and how to repair it

Interaction lens

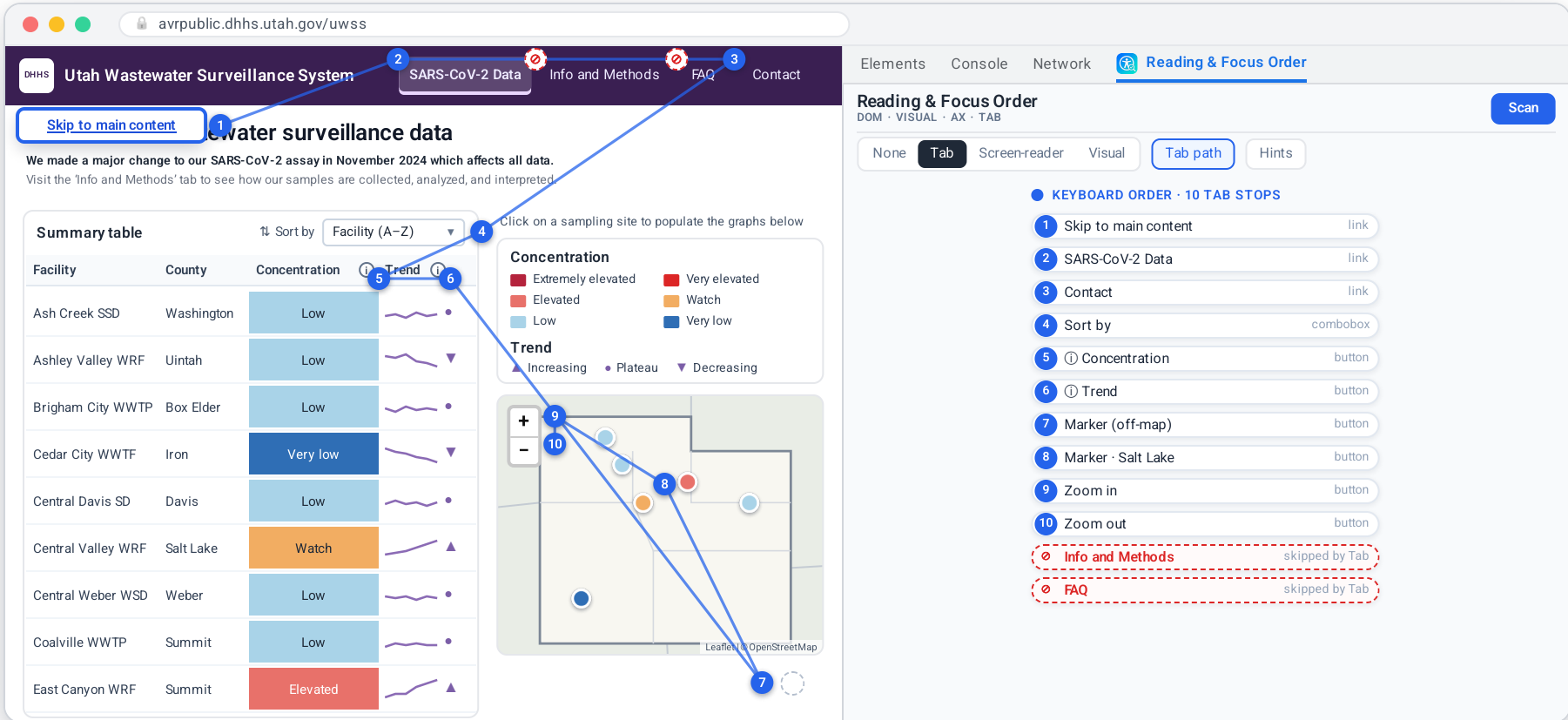
audits every reachable state, not just the first one



4 After a fix, the **run-to-run diff** confirms both zoom-button failures are resolved

Reading & focus order lens

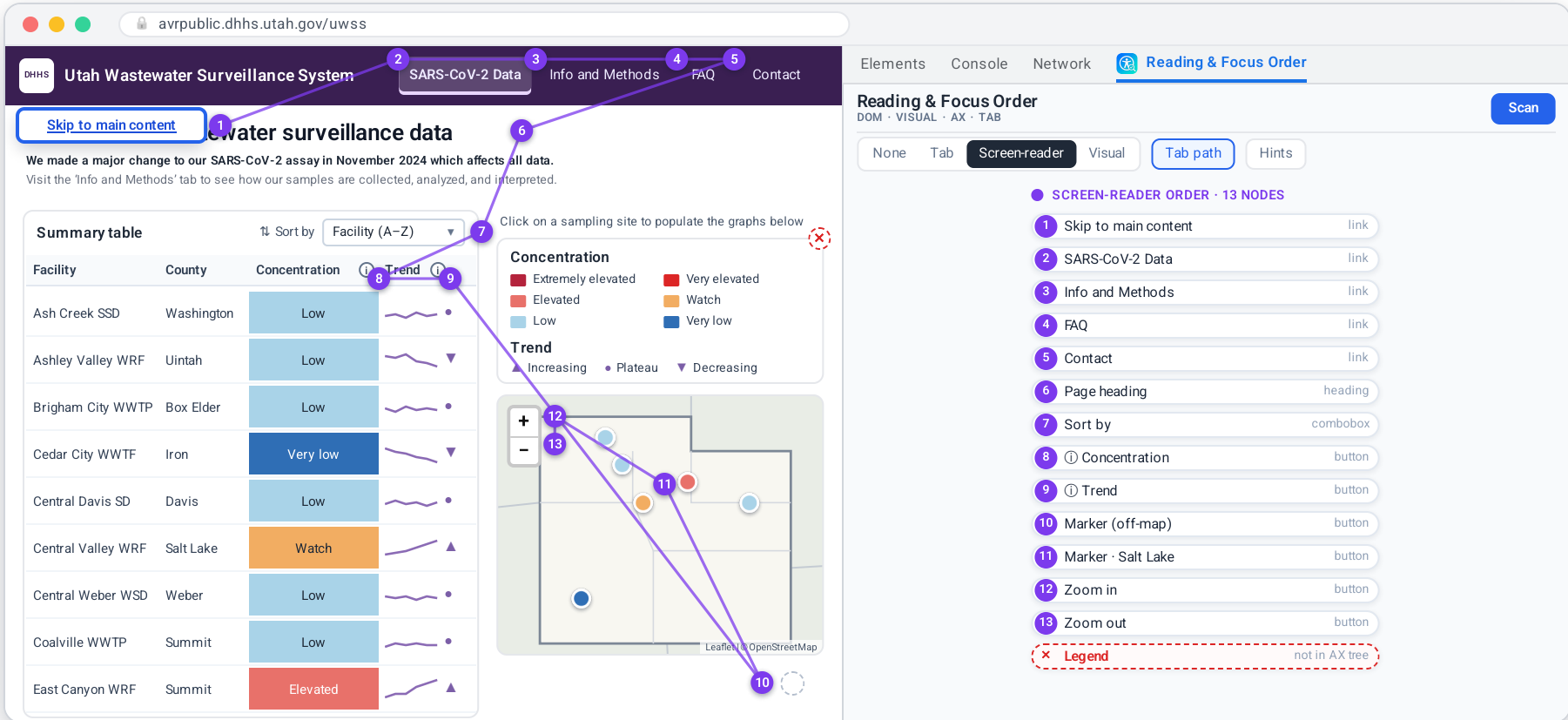
visual · screen reader · keyboard, side by side



1 A **numbered overlay** traces the keyboard (Tab) order in place on the live page

Reading & focus order lens

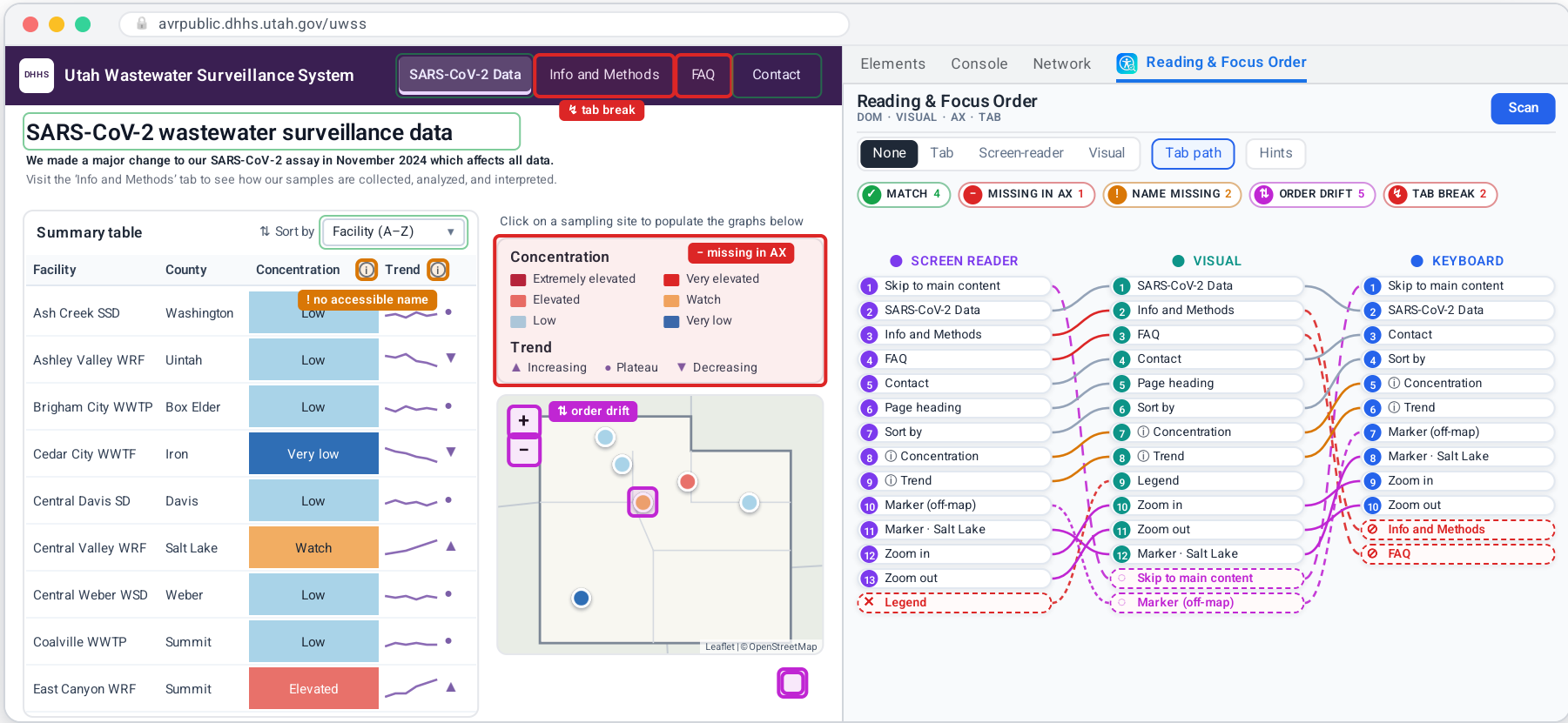
visual · screen reader · keyboard, side by side



2 One switch shows the **screen-reader order** on the same page

Reading & focus order lens

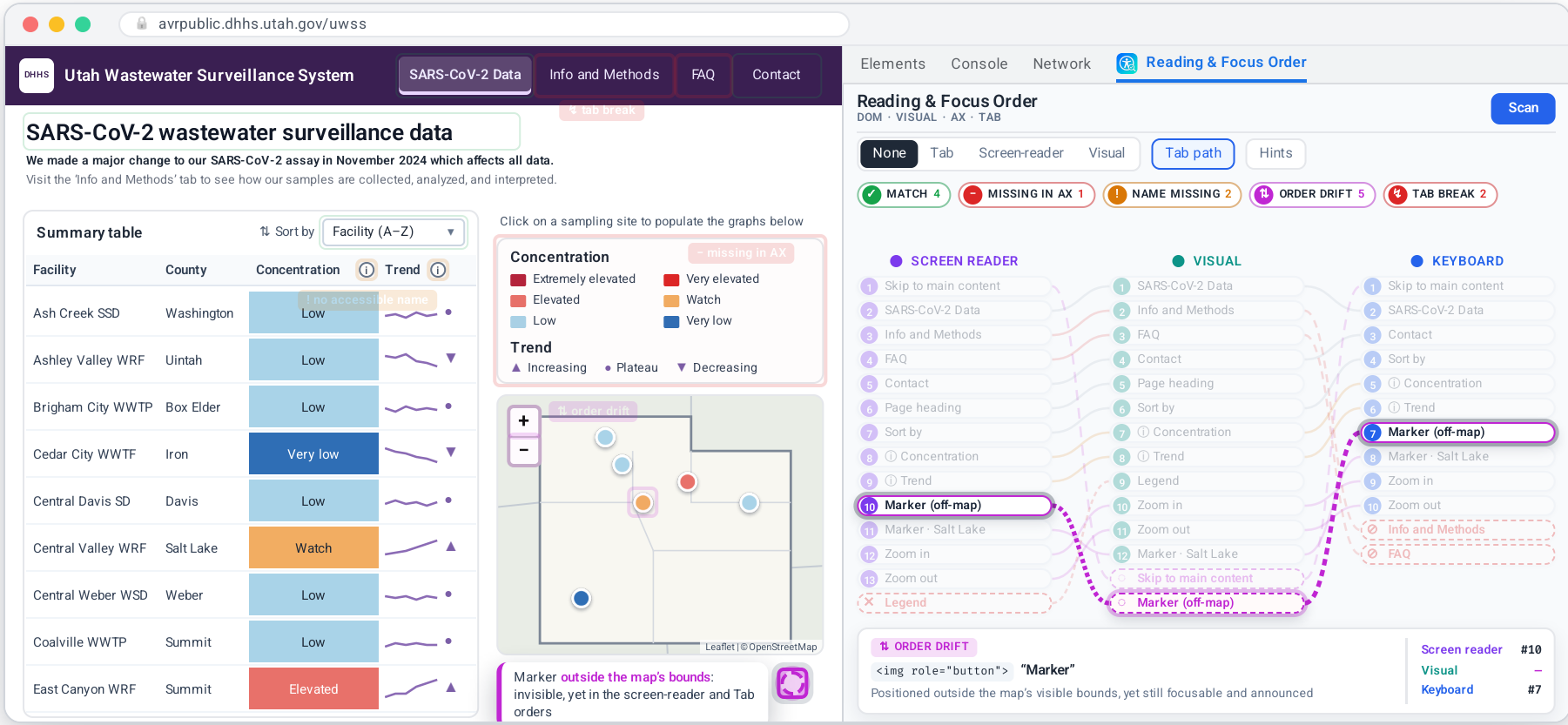
visual · screen reader · keyboard, side by side



3 All three orders are **aligned** side by side, and every divergence is **classified**

Reading & focus order lens

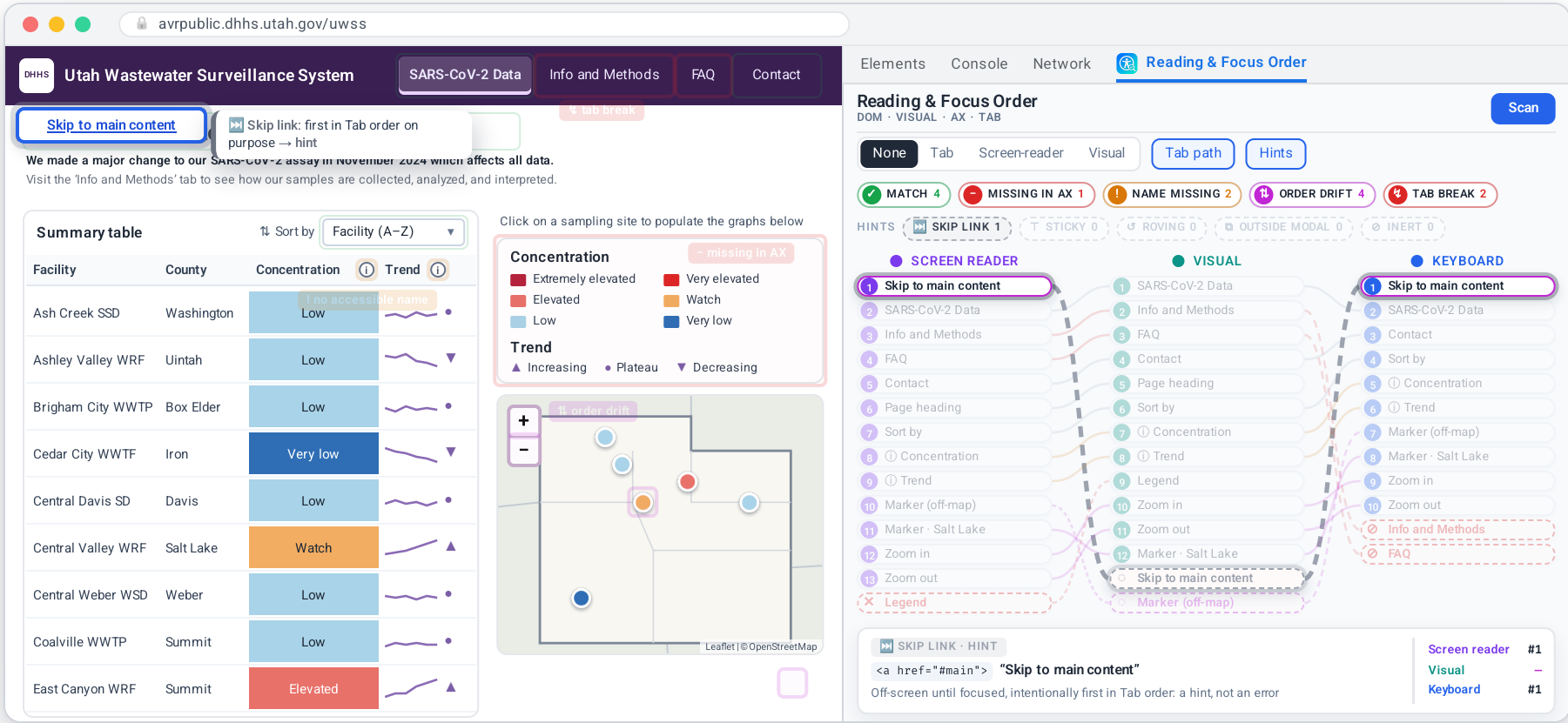
visual · screen reader · keyboard, side by side



4 **Linked views**: select a divergence to locate it on the page and inspect it

Reading & focus order lens

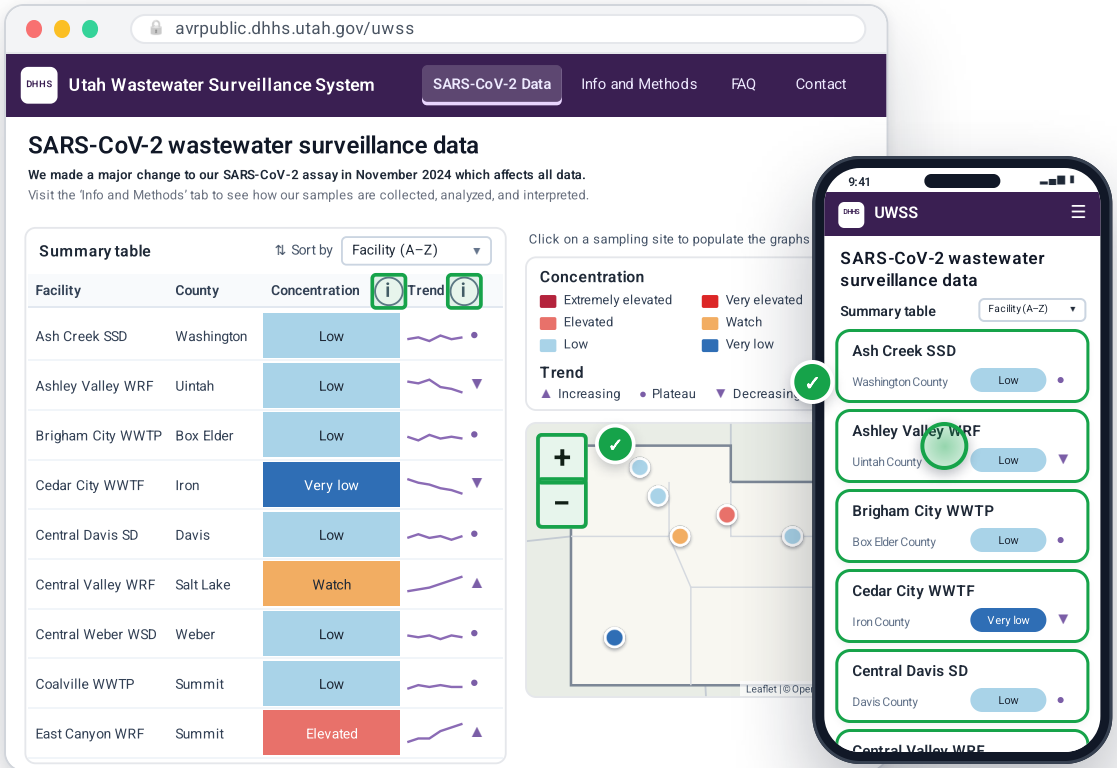
visual · screen reader · keyboard, side by side



5 Deliberate patterns, like skip links, are downgraded to **hints**, not errors

Use case: Utah Wastewater Surveillance System

interaction lens: audit, fix, confirm



✓ Target size

✗ Info icons, sort glyphs, and map zoom controls are too small to hit

✓ **Enlarged targets with uniform shapes**

✓ Mobile table

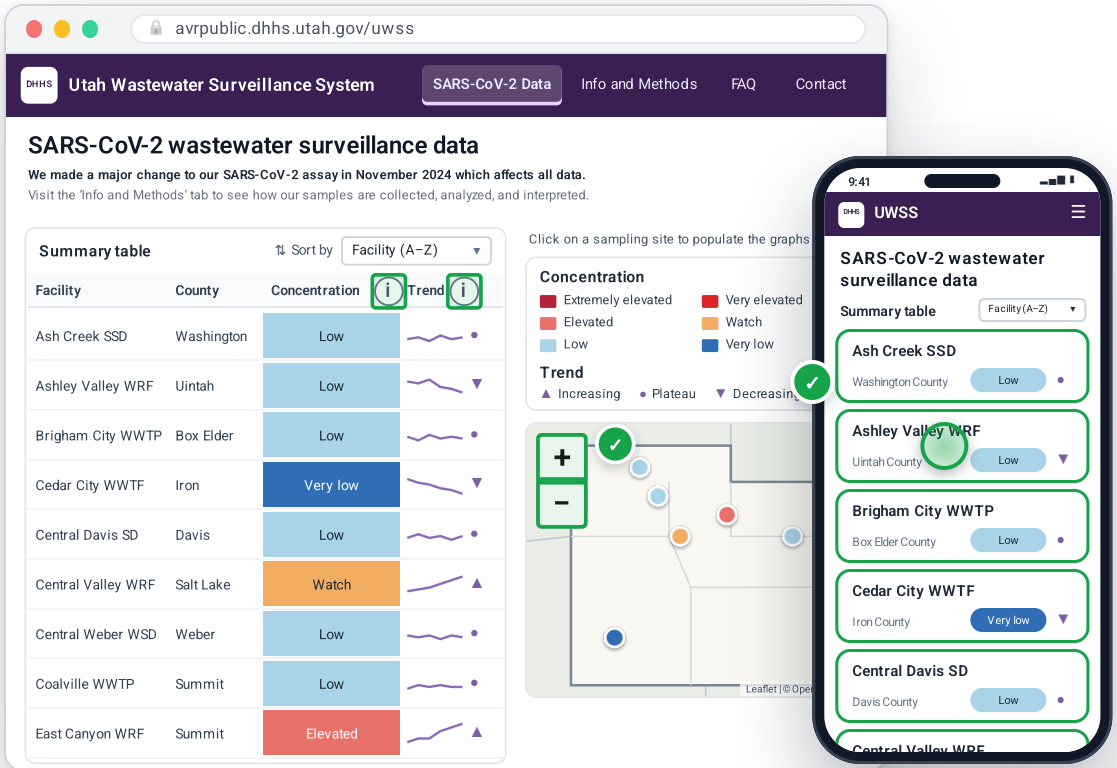
✗ On phones, facility rows collapse too thin to tap

✓ **Responsive card layout on small screens**

History / Diff Diff view confirmed both fixes **cleared** across iterations

Use case: Utah Wastewater Surveillance System

interaction lens: audit, fix, confirm



✓ Target size

✗ Info icons, sort glyphs, and map zoom controls are too small to hit

✓ **Enlarged targets with uniform shapes**

✓ Mobile table

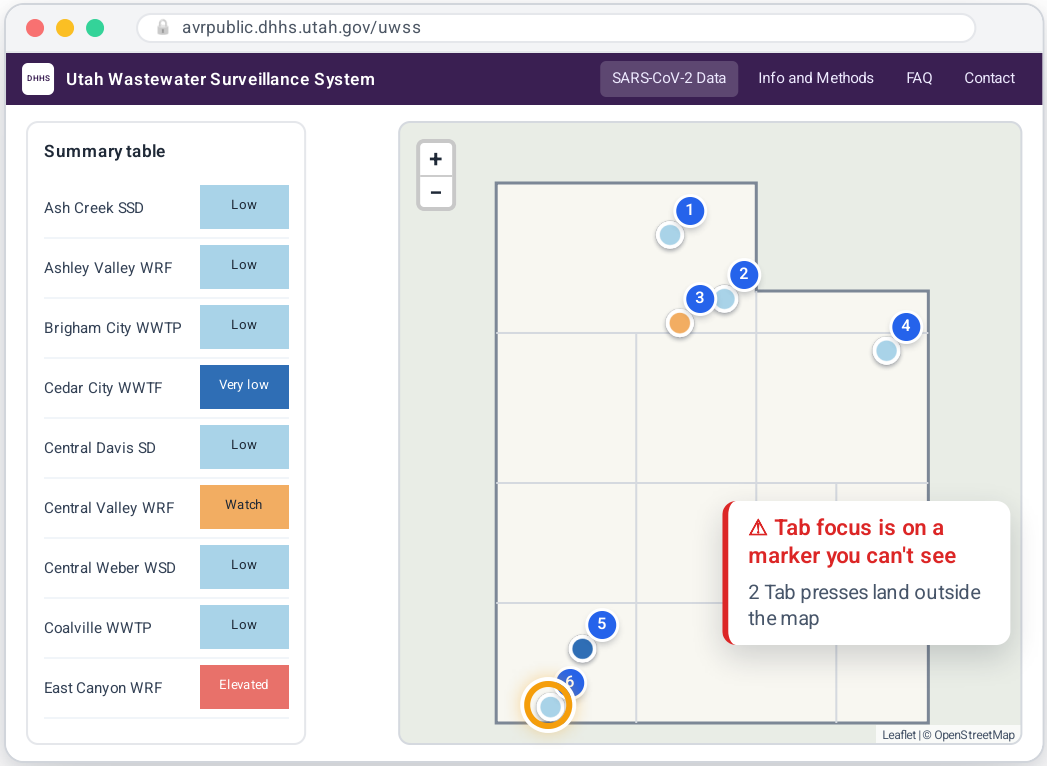
✗ On phones, facility rows collapse too thin to tap

✓ **Responsive card layout on small screens**

History / Diff Diff view confirmed both fixes **cleared** across iterations

Use case: map markers that pan out of view

order lens · UWSS facility map



Reading & Focus Order — MAP MARKERS

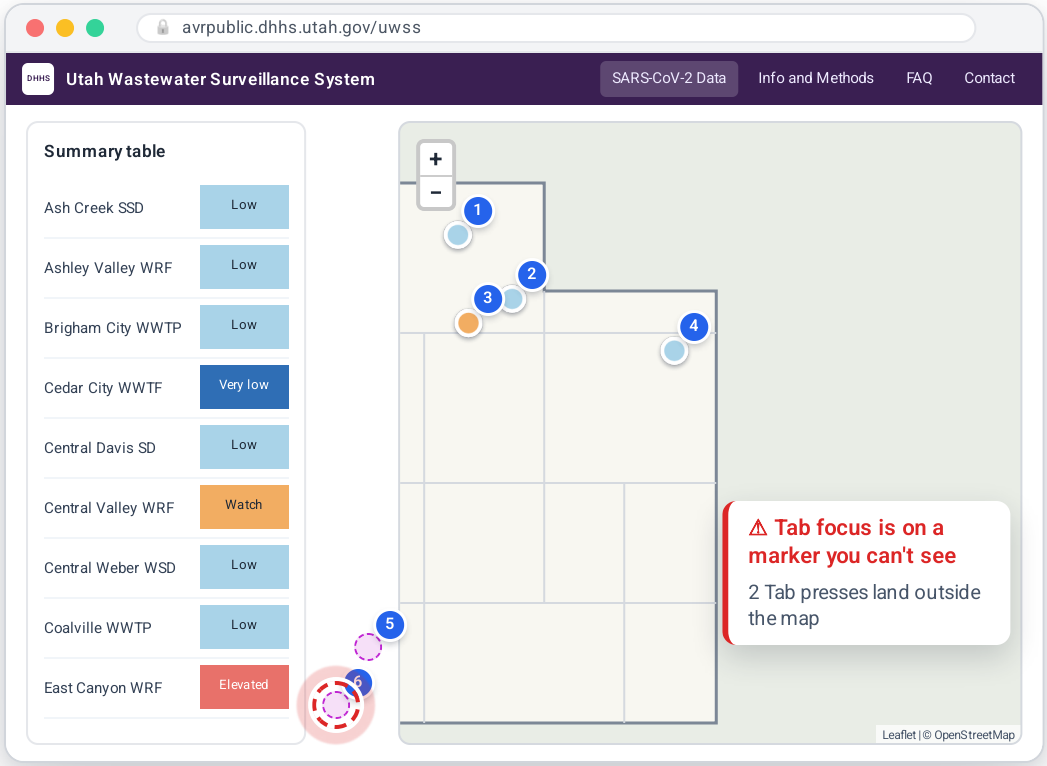
SCREEN READER	VISUAL	KEYBOARD
1 Brigham City	1 Brigham City	1 Brigham City
2 Coalville	2 Coalville	2 Coalville
3 Central Valley	3 Central Valley	3 Central Valley
4 Ashley Valley	4 Ashley Valley	4 Ashley Valley
5 Cedar City	5 Cedar City	5 Cedar City
6 Ash Creek	6 Ash Creek	6 Ash Creek

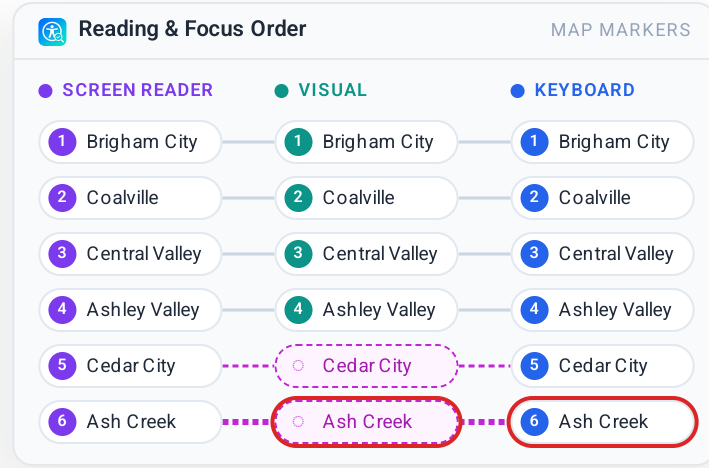
INITIAL LOAD
Visual, screen-reader, and Tab orders match — **PASS**

AFTER PANNING THE MAP
Tab focus lands on markers you can't see — **FAIL**

Use case: map markers that pan out of view

order lens · UWSS facility map



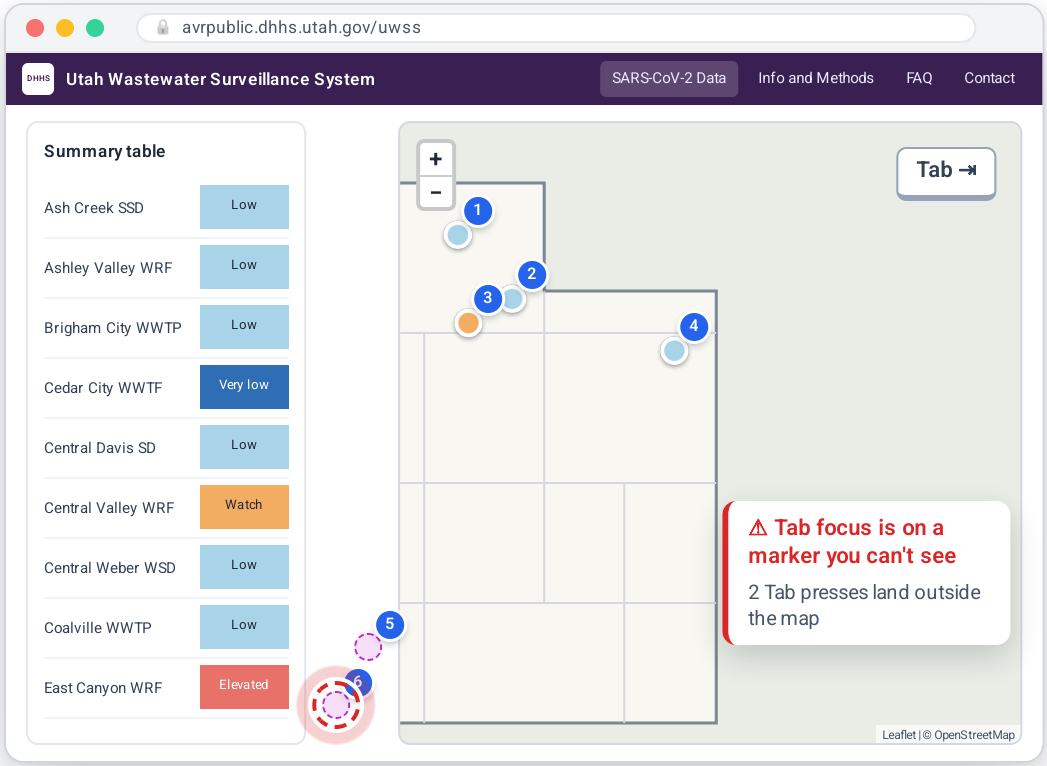


INITIAL LOAD
Visual, screen-reader, and Tab orders match
PASS

AFTER PANNING THE MAP
Tab focus lands on markers you can't see
FAIL

Use case: map markers that pan out of view

order lens · UWSS facility map



Reading & Focus Order — MAP MARKERS

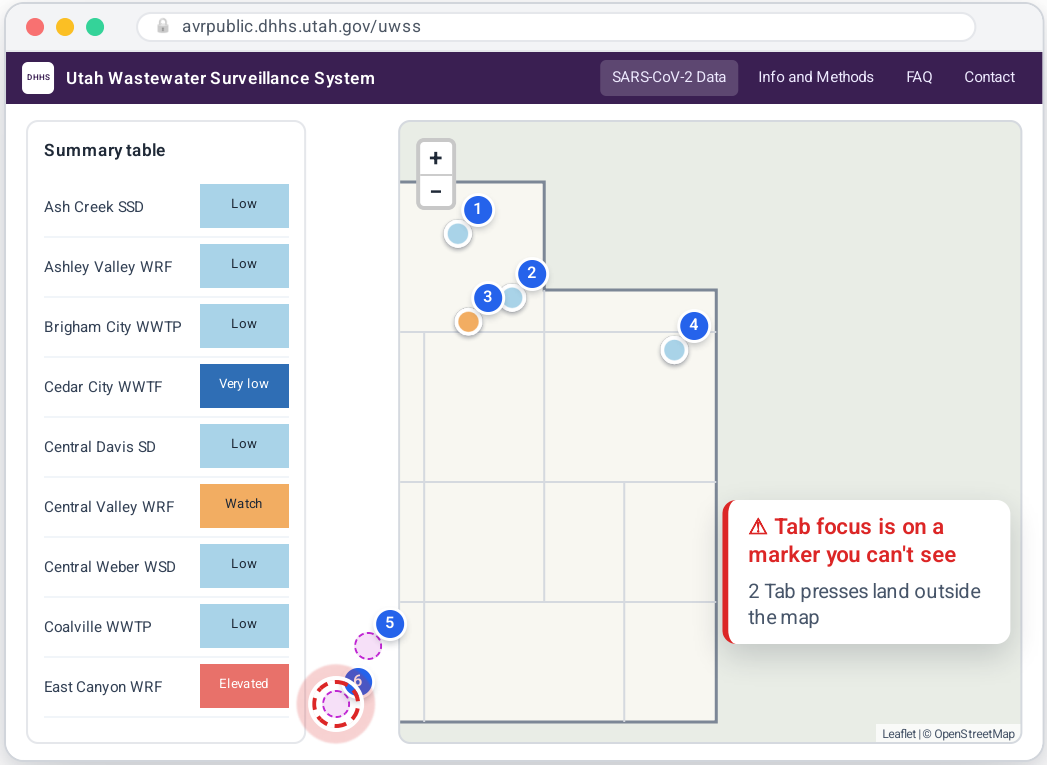
SCREEN READER	VISUAL	KEYBOARD
1 Brigham City	1 Brigham City	1 Brigham City
2 Coalville	2 Coalville	2 Coalville
3 Central Valley	3 Central Valley	3 Central Valley
4 Ashley Valley	4 Ashley Valley	4 Ashley Valley
5 Cedar City	Cedar City	5 Cedar City
6 Ash Creek	Ash Creek	6 Ash Creek

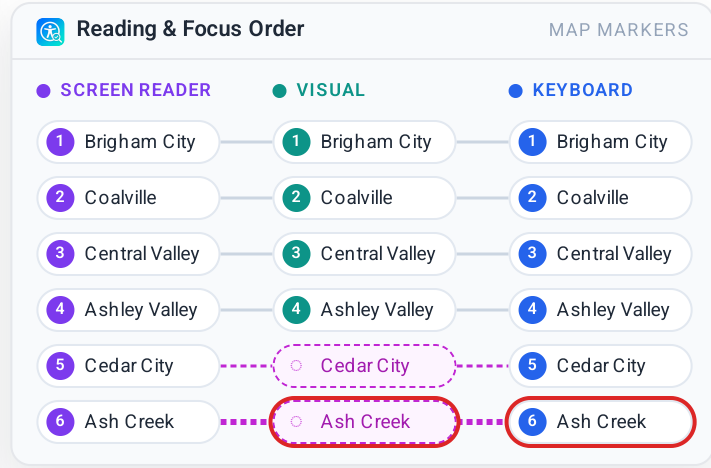
INITIAL LOAD
Visual, screen-reader, and Tab orders match — **PASS**

AFTER PANNING THE MAP
Tab focus lands on markers you can't see — **FAIL**

Use case: map markers that pan out of view

order lens · UWSS facility map





INITIAL LOAD
Visual, screen-reader, and Tab orders match
PASS

AFTER PANNING THE MAP
Tab focus lands on markers you can't see
FAIL

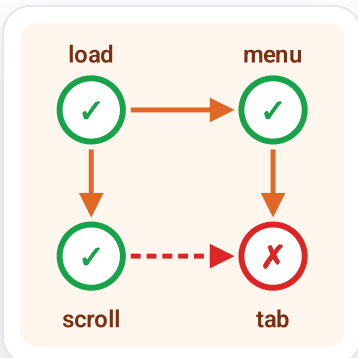


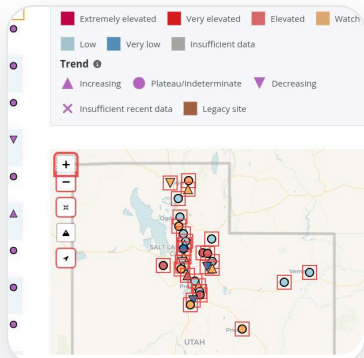
In the wild

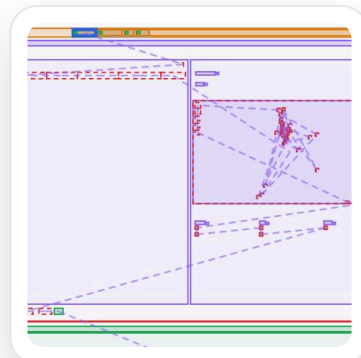
We also ran it on **Our World in Data**, a large public visualization site we don't control

ourworldindata.org/poverty

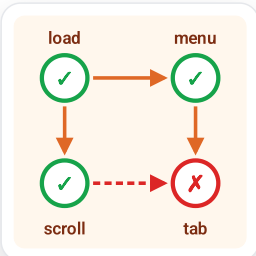
Final Discussion

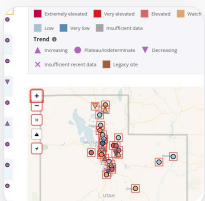


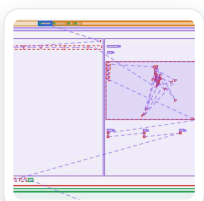




Final Discussion



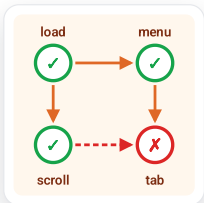




Shift the unit of review

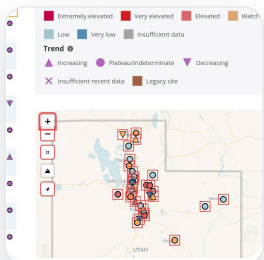
Violations per element in one page state are not enough: accessibility depends on relationships among controls and navigation paths. Audit both the **nodes** and the **edges** of a reactive site's **interaction graph**.

Final Discussion



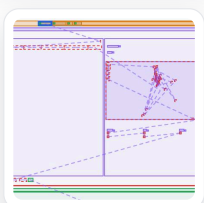
Shift the unit of review

Violations per element in one page state are not enough: accessibility depends on relationships among controls and navigation paths. Audit both the **nodes** and the **edges** of a reactive site's **interaction graph**.

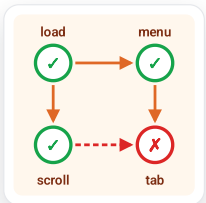


Libraries are the bottleneck

Leaflet.js and **Plotly.js** limited how far keyboard order could be fixed, regardless of author intent. Low-level libraries should expose accessibility hooks, such as **programmatic focus order** for each data point.

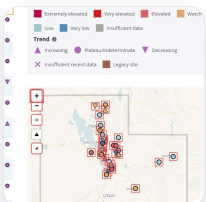


Final Discussion



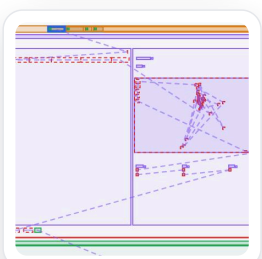
Shift the unit of review

Violations per element in one page state are not enough: accessibility depends on relationships among controls and navigation paths. Audit both the **nodes** and the **edges** of a reactive site's **interaction graph**.



Libraries are the bottleneck

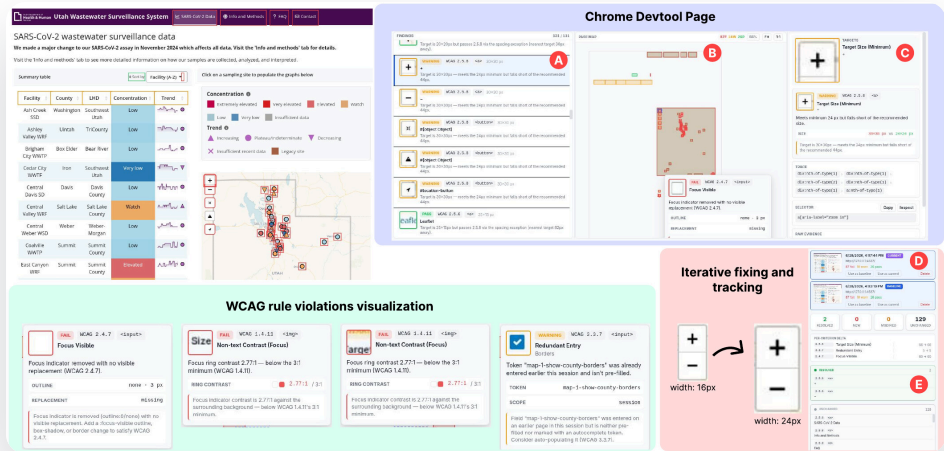
Leaflet.js and **Plotly.js** limited how far keyboard order could be fixed, regardless of author intent. Low-level libraries should expose accessibility hooks, such as **programmatic focus order** for each data point.



Limitations and future work

The interaction lens audits only states its automation can **reach**; very complex interactions may stay unvisited. Reading and focus order is **one piece** of accessibility; next, we will extend the tool to audit **color and visualization accessibility**.

Thank You!



Chrome Web Store

Visual Accessibility Auditing Across Interaction States and Reading Orders

Authors

Rahat Zaman

Jake Wagoner

Tingying He

Alexander Lex

Paul Rosen

Resources

rahatzamancse/site-accessibility-auditor

Site Accessibility Auditor

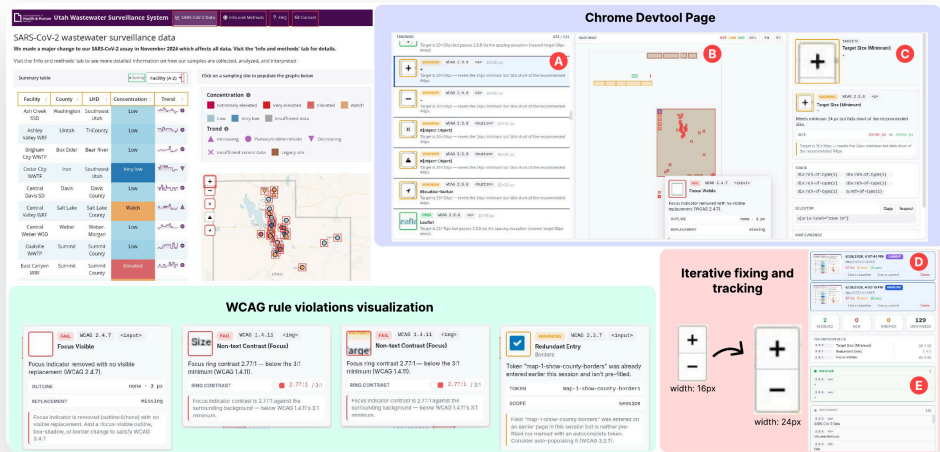
Presenter

Rahat Zaman

rahat.zaman@utah.edu

rahatzaman.me

Thank You!



Chrome Web Store

Try it in Chrome!

Visual Accessibility Auditing Across Interaction States and Reading Orders

Authors

Rahat Zaman

Jake Wagoner

Tingying He

Alexander Lex

Paul Rosen

Resources

rahatzamancse/site-accessibility-auditor

Site Accessibility Auditor

Presenter

Rahat Zaman

rahat.zaman@utah.edu

rahatzaman.me

Other Reference Slides

Positioning against prior work

Rule-based checkers

Lighthouse · axe-core · WAVE · W3C evaluation tools

Routine in development, but automated checks alone leave much of WCAG unexamined, add false positives, and miss barriers blind users meet in practice.

One snapshot · rule list

Dynamic exploration

MAUVE guideline checks on dynamic pages · **Demodocus** reachable-state graph · **QualState** SPA state exploration · **BAGEL** keyboard navigation barriers

Coverage over a state graph, not page-grounded evidence

Making structure visible

aDesigner overlays blind-user problems on pages · **DevTools** inspectors show DOM, AX properties, tab order · **Olli**, **Data Navigator** navigation structures

Views shown separately; structures unseen by sighted authors



Site Accessibility Auditor presents interaction-dependent WCAG 2.2 findings as page-grounded visual evidence, and aligns visual, AX, and keyboard orders in one representation, inside DevTools.

Implementation

Chrome Manifest V3 extension · all analysis runs locally

DevTools panels

Interaction lens · Order lens

⇅ `inspectedWindow.eval`

Inspected page (unmodified)

in-page instrumentation

Service worker
viewport captures

Local storage
audit history

TypeScript · Vite · Svelte 5 · Tailwind CSS · D3

1. **DevTools panels**
   Svelte 5 and Tailwind CSS; coordinated SVG views drawn with D3: containment tree, drift ribbon, state graph, page maps
2. **Inspected-window evaluation**
   Analysis executes inside the live, unmodified page through the DevTools inspected-window API
3. **In-page instrumentation**
   One traversal yields the AX, visual, and keyboard orders with geometry and layout signals; a breadth-first crawler explores states with fingerprints and rollback
4. **Background service worker**
   Captures the viewport while scrolling and stitches a full-page image, so every finding has stable page coordinates
5. **Local storage**
   Findings signed by criterion, category, element trace, and evidence hash power the new / resolved / persistent diff

Implementation

Chrome Manifest V3 extension · all analysis runs locally

DevTools panels

Interaction lens · Order lens

⇅ `inspectedWindow.eval`

Inspected page (unmodified)

in-page instrumentation

Service worker
viewport captures

Local storage
audit history

TypeScript · Vite · Svelte 5 · Tailwind CSS · D3

1. **DevTools panels**
   Svelte 5 and Tailwind CSS; coordinated SVG views drawn with D3: containment tree, drift ribbon, state graph, page maps
2. **Inspected-window evaluation**
   Analysis executes inside the live, unmodified page through the DevTools inspected-window API
3. **In-page instrumentation**
   One traversal yields the AX, visual, and keyboard orders with geometry and layout signals; a breadth-first crawler explores states with fingerprints and rollback
4. **Background service worker**
   Captures the viewport while scrolling and stitches a full-page image, so every finding has stable page coordinates
5. **Local storage**
   Findings signed by criterion, category, element trace, and evidence hash power the new / resolved / persistent diff

Implementation

Chrome Manifest V3 extension · all analysis runs locally

DevTools panels

Interaction lens · Order lens

⇅ `inspectedWindow.eval`

Inspected page (unmodified)

in-page instrumentation

Service worker
viewport captures

Local storage
audit history

TypeScript · Vite · Svelte 5 · Tailwind CSS · D3

1. **DevTools panels**
   Svelte 5 and Tailwind CSS; coordinated SVG views drawn with D3: containment tree, drift ribbon, state graph, page maps
2. **Inspected-window evaluation**
   Analysis executes inside the live, unmodified page through the DevTools inspected-window API
3. **In-page instrumentation**
   One traversal yields the AX, visual, and keyboard orders with geometry and layout signals; a breadth-first crawler explores states with fingerprints and rollback
4. **Background service worker**
   Captures the viewport while scrolling and stitches a full-page image, so every finding has stable page coordinates
5. **Local storage**
   Findings signed by criterion, category, element trace, and evidence hash power the new / resolved / persistent diff

Implementation

Chrome Manifest V3 extension · all analysis runs locally

DevTools panels

Interaction lens · Order lens

⇅ `inspectedWindow.eval`

Inspected page (unmodified)

in-page instrumentation

Service worker
viewport captures

Local storage
audit history

TypeScript · Vite · Svelte 5 · Tailwind CSS · D3

1. **DevTools panels**
   Svelte 5 and Tailwind CSS; coordinated SVG views drawn with D3: containment tree, drift ribbon, state graph, page maps
2. **Inspected-window evaluation**
   Analysis executes inside the live, unmodified page through the DevTools inspected-window API
3. **In-page instrumentation**
   One traversal yields the AX, visual, and keyboard orders with geometry and layout signals; a breadth-first crawler explores states with fingerprints and rollback
4. **Background service worker**
   Captures the viewport while scrolling and stitches a full-page image, so every finding has stable page coordinates
5. **Local storage**
   Findings signed by criterion, category, element trace, and evidence hash power the new / resolved / persistent diff

Implementation

Chrome Manifest V3 extension · all analysis runs locally

DevTools panels

Interaction lens · Order lens

⇅ `inspectedWindow.eval`

Inspected page (unmodified)

in-page instrumentation

Service worker
viewport captures

Local storage
audit history

TypeScript · Vite · Svelte 5 · Tailwind CSS · D3

1 **DevTools panels**
Svelte 5 and Tailwind CSS; coordinated SVG views drawn with D3: containment tree, drift ribbon, state graph, page maps

2 **Inspected-window evaluation**
Analysis executes inside the live, unmodified page through the DevTools inspected-window API

3 **In-page instrumentation**
One traversal yields the AX, visual, and keyboard orders with geometry and layout signals; a breadth-first crawler explores states with fingerprints and rollback

4 **Background service worker**
Captures the viewport while scrolling and stitches a full-page image, so every finding has stable page coordinates

5 **Local storage**
Findings signed by criterion, category, element trace, and evidence hash power the new / resolved / persistent diff

Implementation

Chrome Manifest V3 extension · all analysis runs locally

DevTools panels

Interaction lens · Order lens

⇅ `inspectedWindow.eval`

Inspected page (unmodified)

in-page instrumentation

Service worker
viewport captures

Local storage
audit history

TypeScript · Vite · Svelte 5 · Tailwind CSS · D3

1 **DevTools panels**
Svelte 5 and Tailwind CSS; coordinated SVG views drawn with D3: containment tree, drift ribbon, state graph, page maps

2 **Inspected-window evaluation**
Analysis executes inside the live, unmodified page through the DevTools inspected-window API

3 **In-page instrumentation**
One traversal yields the AX, visual, and keyboard orders with geometry and layout signals; a breadth-first crawler explores states with fingerprints and rollback

4 **Background service worker**
Captures the viewport while scrolling and stitches a full-page image, so every finding has stable page coordinates

5 **Local storage**
Findings signed by criterion, category, element trace, and evidence hash power the new / resolved / persistent diff