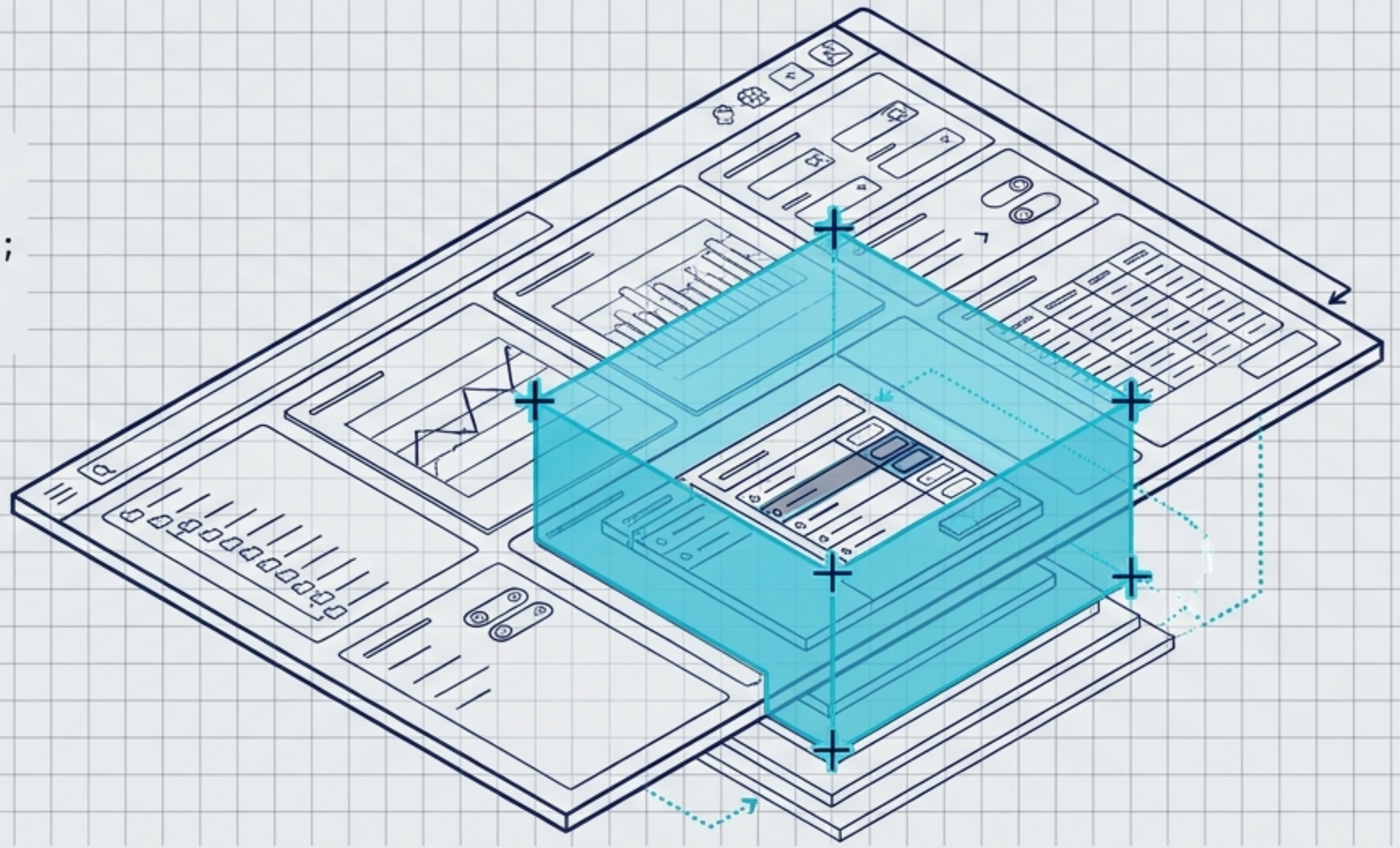
DIGITAL ACCESSIBILITY: S10E10

Companion Lab: Focus Management and Roving Tabindex

Architecture & Engineering Standards | The Operations Desk Blueprint

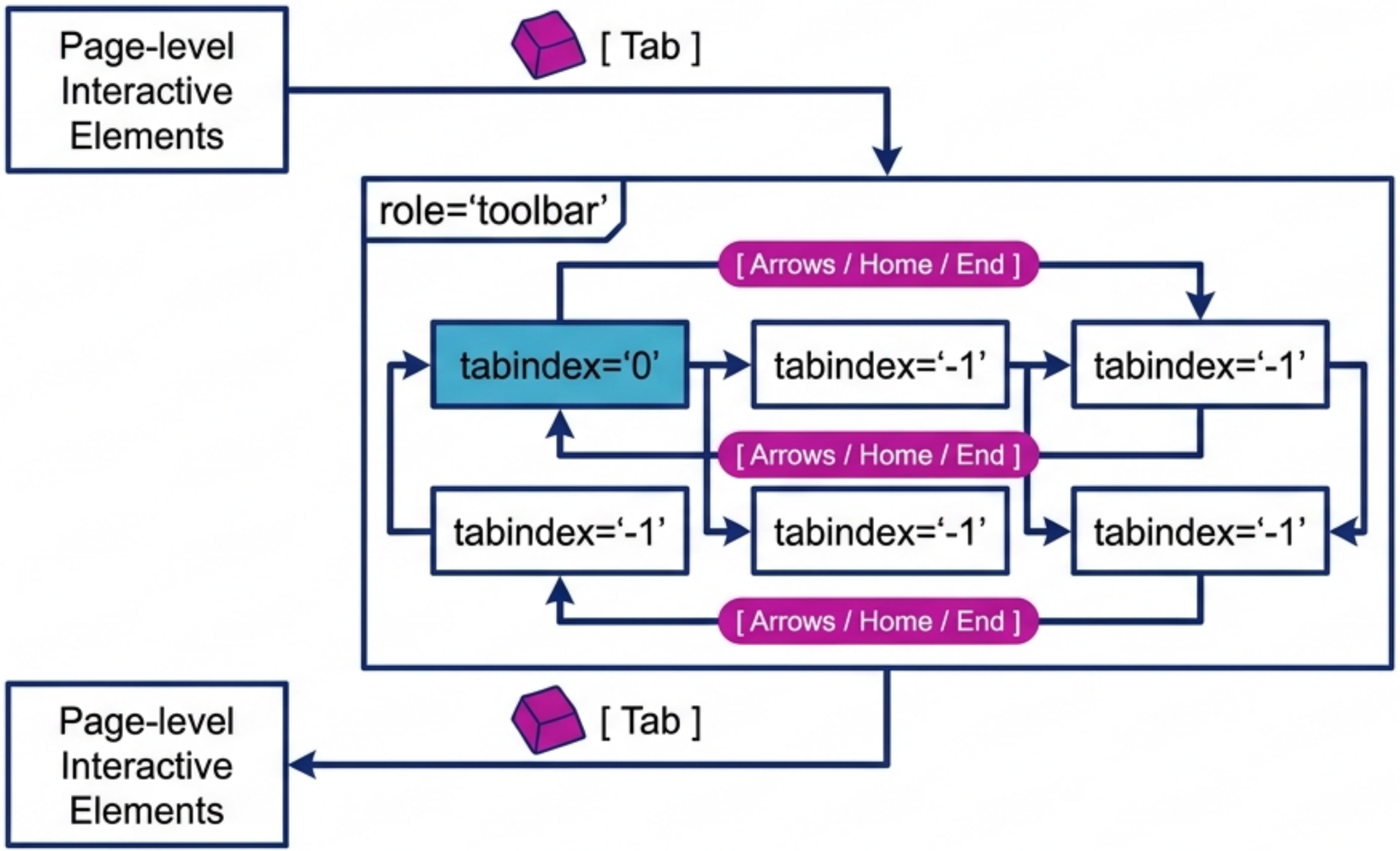
PREVIOUSLY ON S10E09:
We scaled from individual accessible primitives to a page-wide focus architecture; now we master complex widget widget state and precision focus routing.



Gemini Notebook

1. Single-Tab-Stop Composite Widgets

Codifying the Single-Tab-Stop Contract



The Problem:
90 sequential interactive elements create a keyboard trap of exhaustion.

The Contract (APG):
Treat composite widgets as a single interactive ecosystem.

Golden Rule:
Exactly one tabindex='0' (or > 0) active within the composite. All other interactive children must be strictly tabindex='-1'.

Gemini Notebook

1. Single-Tab-Stop Composite Widgets

DOM Focus vs. Virtual Focus

Roving Tabindex (DOM Focus)

```
<div role="toolbar">
  <ul>
    <li id="item1" tabindex="-1">Item 1</li>
    <li id="item2" tabindex="0" class="active">Item 2</li>
    <li id="item3" tabindex="-1">Item 3</li>
  </ul>
</div>
```

[Arrow Right / Down] ->

```
<div role="toolbar">
  <ul>
    <li id="item1" tabindex="-1">Item 1</li>
    <li id="item2" tabindex="-1">Item 2</li>
    <li id="item3" tabindex="0" class="active">Item 3</li>
  </ul>
</div>
```

- **Mechanism:** Physical DOM focus moves.
- **Best for: role='toolbar'** and **role='tablist'**.
- **Key Engineering:** Must remember the last active child on re-entry. Must explicitly invert arrow logic for RTL (**dir='rtl'**) layouts.

aria-activedescendant (Virtual Focus)

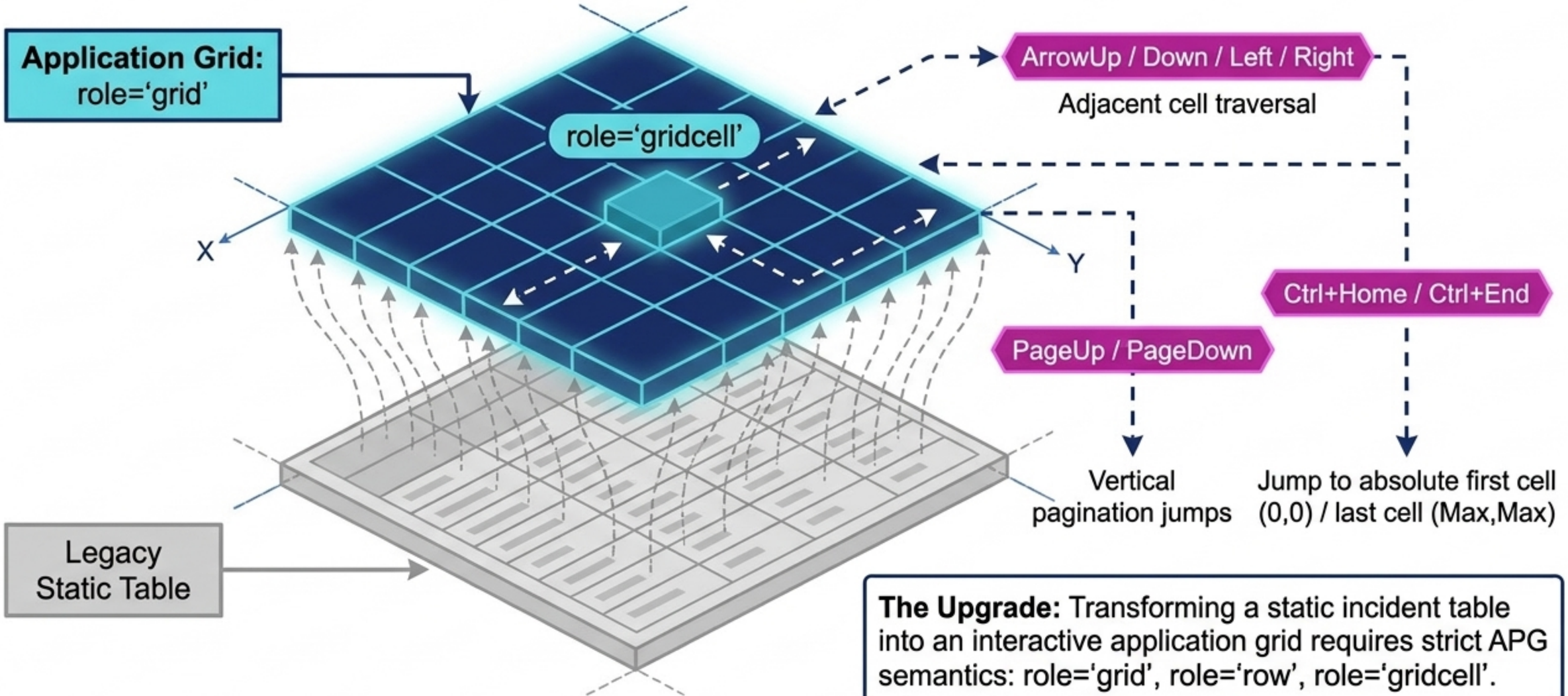
```
<div role="combobox" aria-expanded="true"
aria-haspopup="listbox">
  <input type="text" id="combo-input"
aria-activedescendant="option2">
  <ul role="listbox" id="listbox1">
    <li role="option" id="option1">Option 1</li>
    <li role="option" id="option2" class="active-
descendant">Option 2</li>
    <li role="option" id="option3">Option 3</li>
  </ul>
</div>
```

- **Mechanism:** DOM focus is locked; ARIA points to a "virtually" focused child ID.
- **Best for:** Comboboxes/Filters (where user must keep typing).
- **Key Engineering:** Requires explicit CSS styling for the active descendant, as the native **:focus** pseudo-class won't trigger.

Gemini Notebook

2. Actionable 2D Data Grids

From Static Cohorts to 2D Spatial Navigation



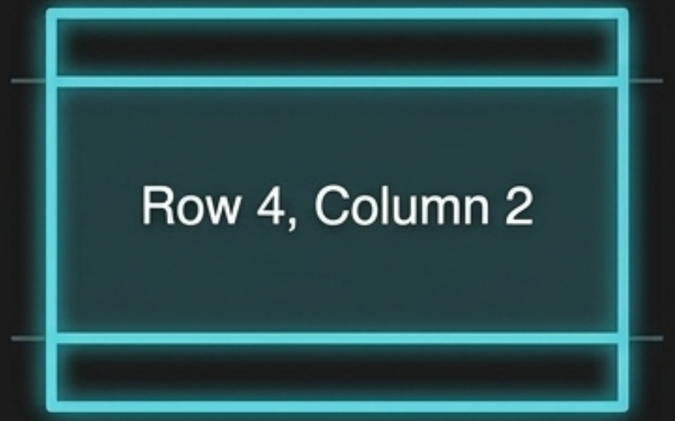
The Upgrade: Transforming a static incident table into an interactive application grid requires strict APG semantics: role='grid', role='row', role='gridcell'.

Gemini Notebook

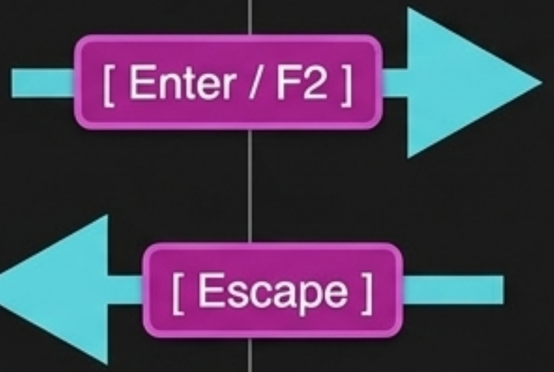
2. Actionable 2D Data Grids

Escaping the WCAG 2.1.2 Keyboard Trap

Cell Navigation Mode



- **Focus** acts on the gridcell.
- **Arrows** move between cells.
- Screen reader announces coordinates: “Row 4, Column 2”.



In-Cell Edit Mode



- Focus moves inside the cell to embedded interactive elements.
- Grid arrow navigation is temporarily suspended.

The Critical Escape: Pressing Escape must instantly return focus to the parent gridcell and restore spatial routing, explicitly preventing a WCAG 2.1.2 Keyboard Trap.

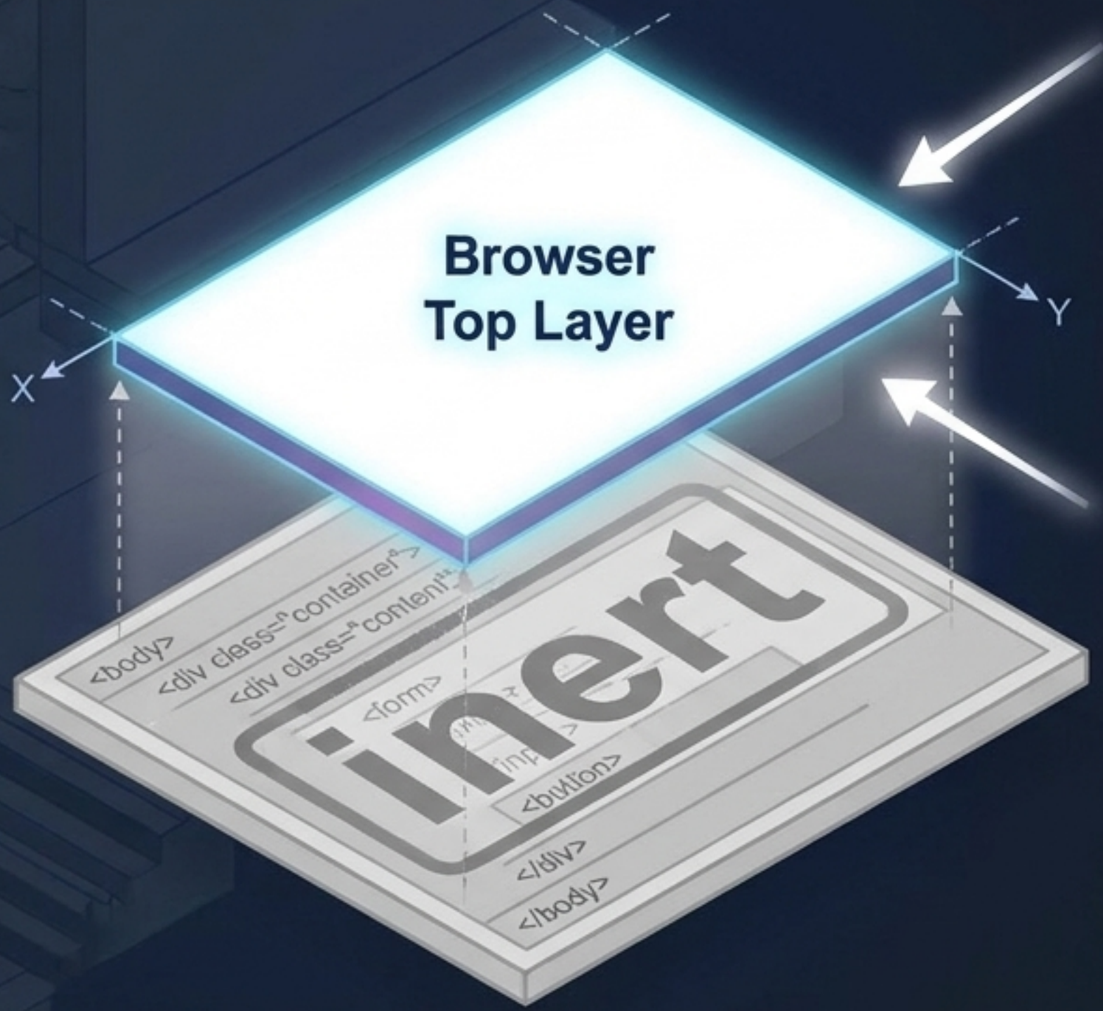
Gemini Notebook

3. Top-Layer Overlays

Deprecating Legacy JS Tab-Traps

```
// Legacy Event Listeners
function trapFocus(e) {
  if (e.key === 'Tab') {
    const focusable =
getFo leElements(m );
    if iftKey && ment.
activeEl t === f able[0]) {
      e.pr tDefa );
      focusa f able.length -
1].focus();
    } else if hiftKey &&
    document lement ===
focusable[f ab ngth - 1]) {
      e.pr tDefa );
      fo ble[0].f ();
    }
  }
}
// ... z-index calculations
// ... activeElement tracking
```

Requires manual keydown traps, active element tracking, and z-index wars.



Native `<dialog>.showModal()`

Automatically promotes to the Top Layer.

Natively traps focus and renders the background DOM **inert**—zero polyfills required.

Light-Dismiss Popovers

Invoked declaratively via

```
<button command='show-modal'
 commandfor='my-popover'>
```

Non-modal.

Enables “light dismiss” (clicking outside or Escape) natively without window event listeners.

Gemini Notebook

3. Top-Layer Overlays

Where Should Initial Focus Land?

Form Modals



Focus the First Input Field

Allows users to immediately begin typing without extra tab strokes.

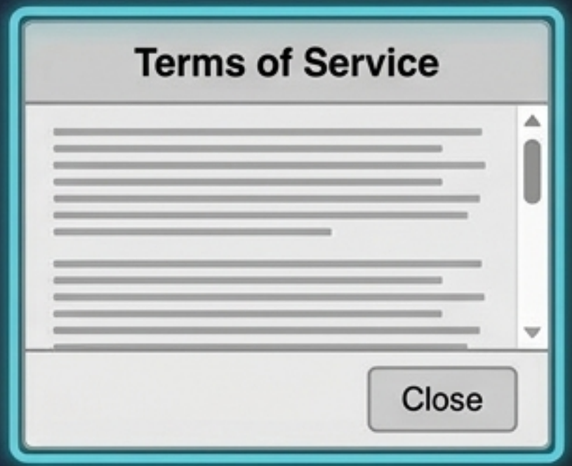
Destructive Confirmations



Focus the Cancel Button

Alert Dialog pattern. Prevents catastrophic accidental deletions if the user double-taps Space or Enter.

Scrollable Content Dialogs



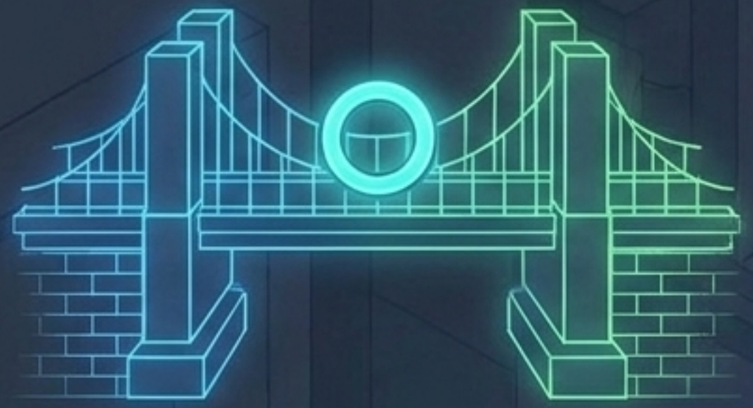
Focus the Dialog Container / Heading

Requires `tabindex='-1'` on the container. Prevents the container from instantly scrolling past the title when targeting the first interactive element.

Gemini Notebook

4. Deterministic Focus Fallback

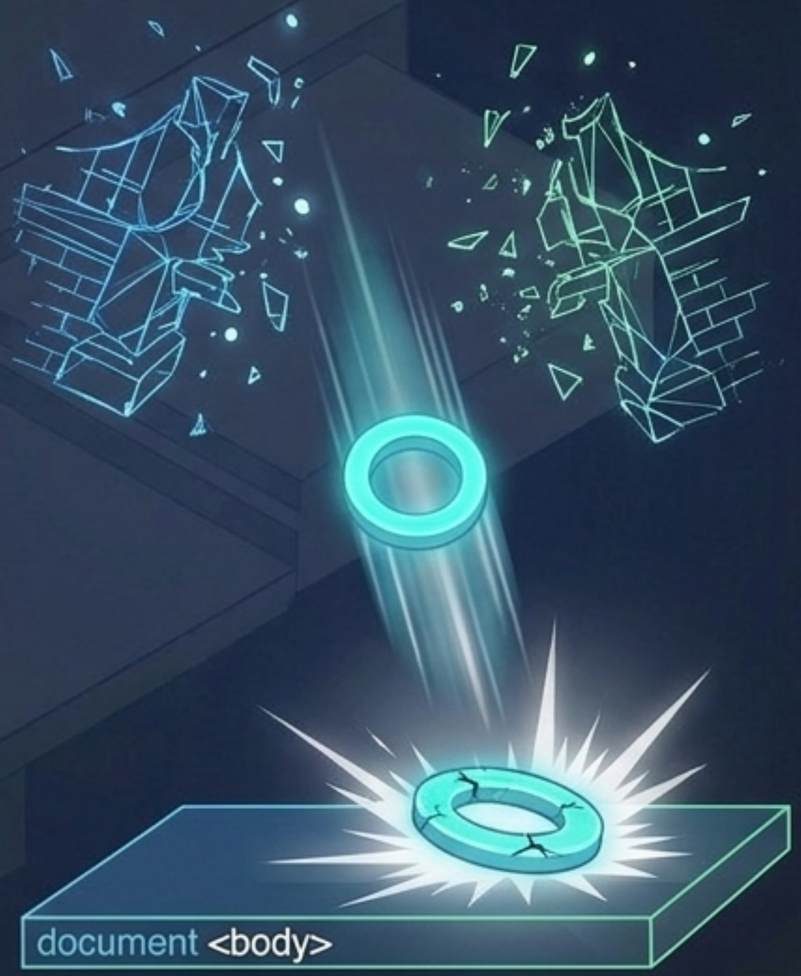
When Triggers Unmount, Focus Dies



User triggers action on “Delete Incident #1042”.

Row is removed from the DOM. triggerRef.current is destroyed.

Focus drops entirely. Next Tab press forces a top-of-page restart.



Summary Box

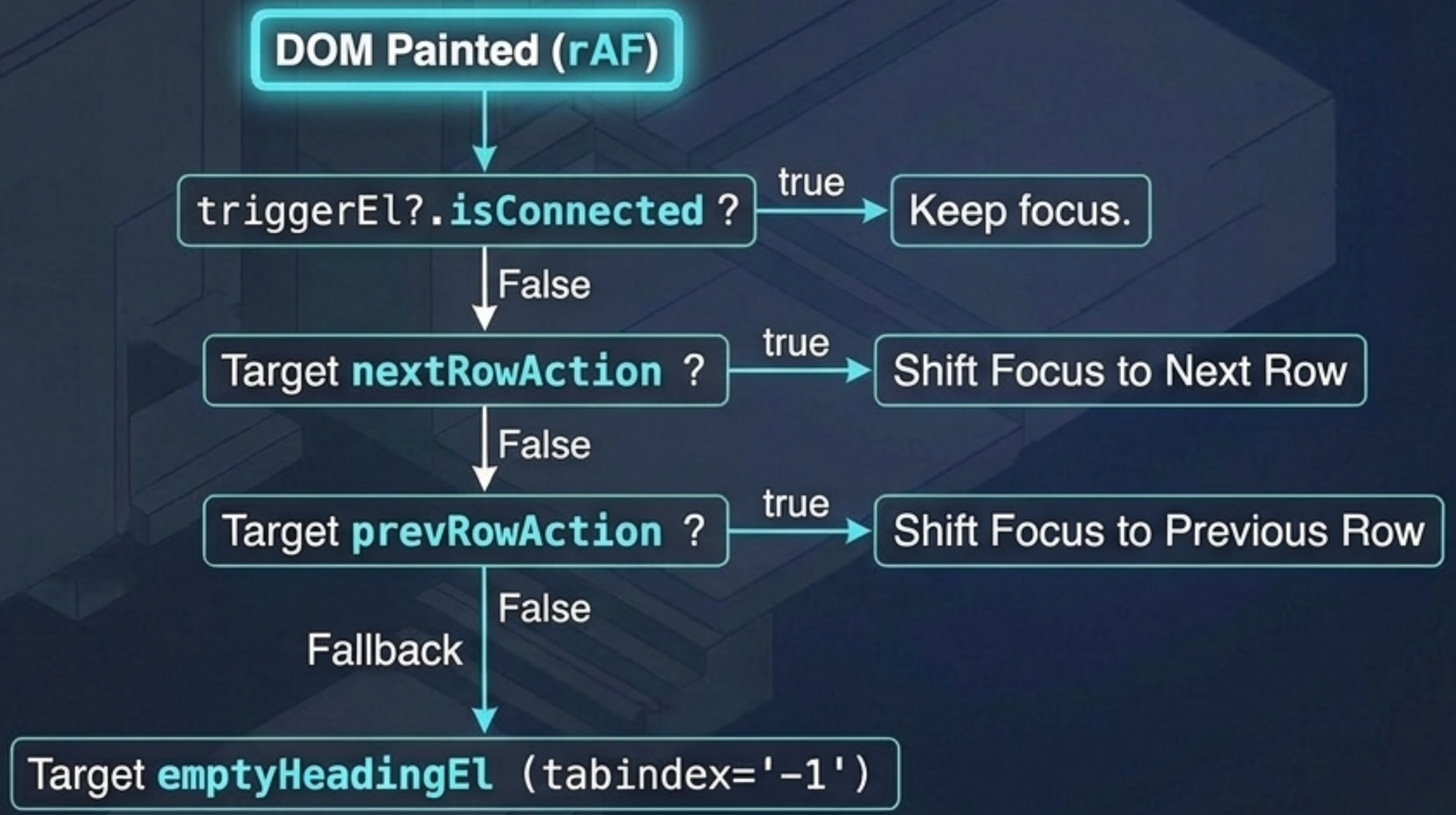
- **The Bug:** Focus has nowhere to anchor.
- **The Consequence:** The browser drops focus entirely, resetting to the document body.
- **The Impact:** Severe accessibility failure. The user hits Tab and lands on “Skip to main content”.

Gemini Notebook

4. Deterministic Focus Fallback

State-Machine-Precise Focus Restoration

The Fix: Calculate the fallback target before the trigger unmounts. Execute the focus shift after the DOM paints.



Execution:

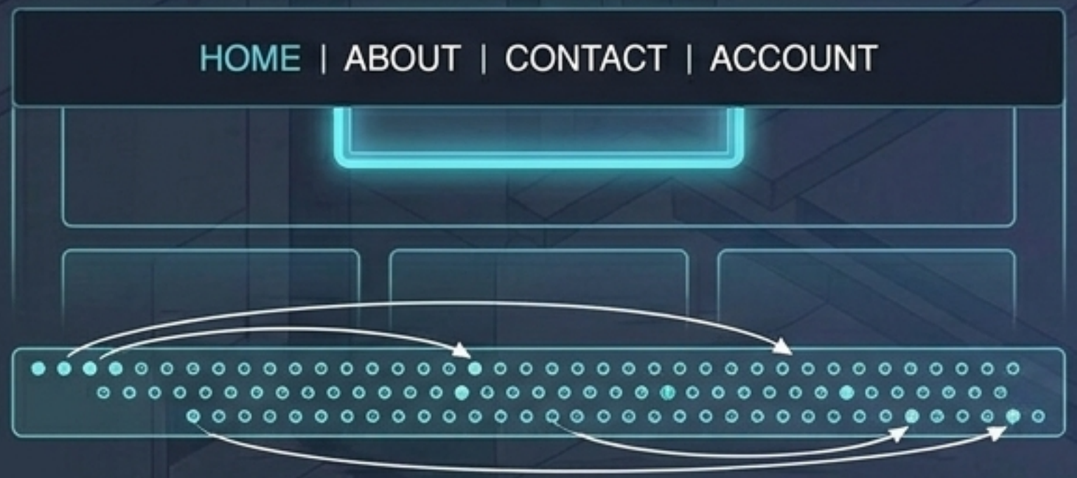
Wrap the focus shift in `requestAnimationFrame` to ensure React reconciliation is complete and the fallback node physically exists in the DOM.

Gemini Notebook

5. 5-Modality Audit (Part 1)

Fixing Visual Burial & Tab Exhaustion

Broken Experience



Visual: Focus completely hidden underneath sticky headers.

Keyboard: 90 sequential tab stops through card toolbars.

Fixed Architecture



Visual: Enforced scroll-margin-top for viewport clearance.

Dual-Band Focus: `:focus-visible` guarantees 3:1 WCAG contrast against any background color.

Keyboard: 90 sequential tab stops collapsed into 10 single-stop interactive areas using roving `tabindex`.

Gemini Notebook

5. 5-Modality Audit (Part 2)

Correcting Verbatim Speech & Layout

Screen Reader Speech

Broken



[Delete confirmed] ... [silence; focus lost] ...
[next Tab: Skip to main content]

Fixed



Alert: Incident 1042 deleted. Actions for Incident 1043:
Database timeout, menu button, collapsed, row 4 of 18.

400% Zoom



112px wrapped sticky header tracked dynamically via `ResizeObserver` and injected to CSS via `--sticky-header-block-size`.

Mobile / Switch / Voice



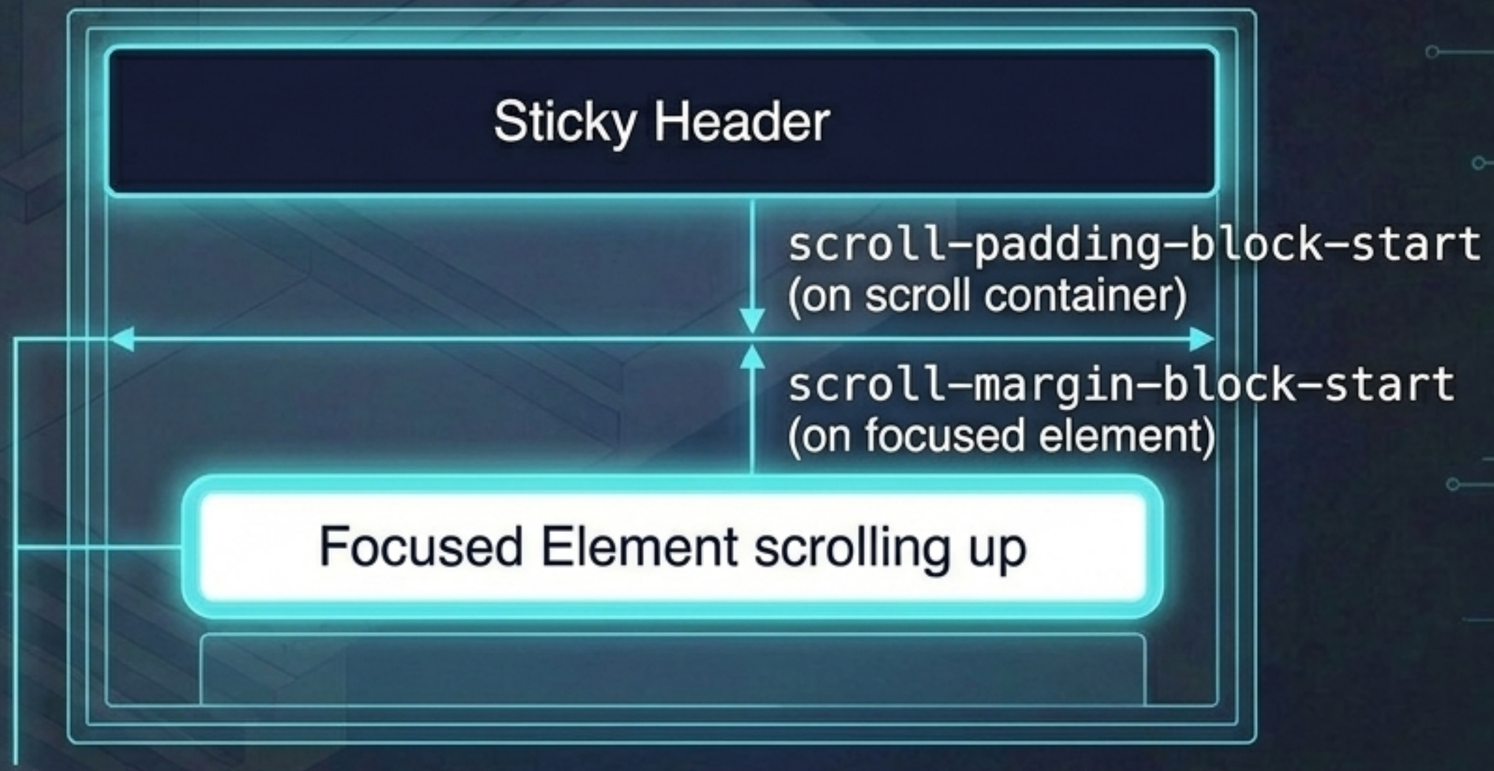
1-scan switch toolbar grouping and strict inert drawer containment prevents invisible background interactions.

Gemini Notebook

6. Sticky Viewport Clearance, SPA & Shadow DOM

Defeating the Sticky Viewport Collision

The Standard: WCAG 2.4.11 (Focus Not Obscured) requires focused elements are never fully hidden by author-created content.



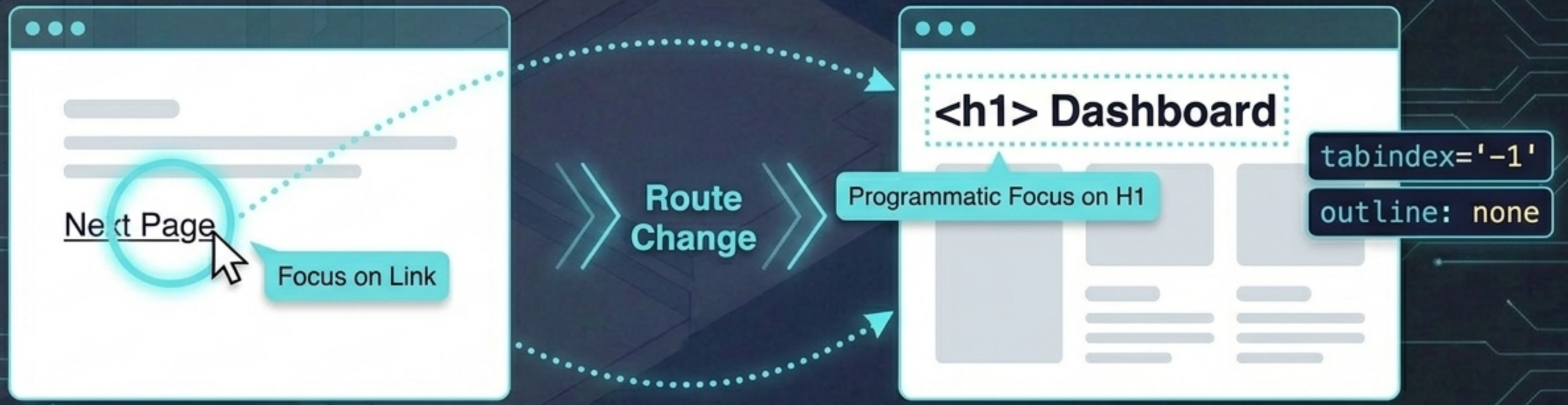
The CSS Implementation: Combine both properties, dynamically updating them via custom properties tied to the header's calculated height to guarantee mathematical clearance.

Gemini Notebook

6. Sticky Viewport Clearance, SPA & Shadow DOM

Managing Focus Across SPA Routing

The Issue: Single Page Applications (SPAs) intercept routing, meaning they do not trigger native browser page-load focus resets.



The Standard Fix: Programmatically move focus to the primary <h1> (given **tabindex='-1'**) on route change.

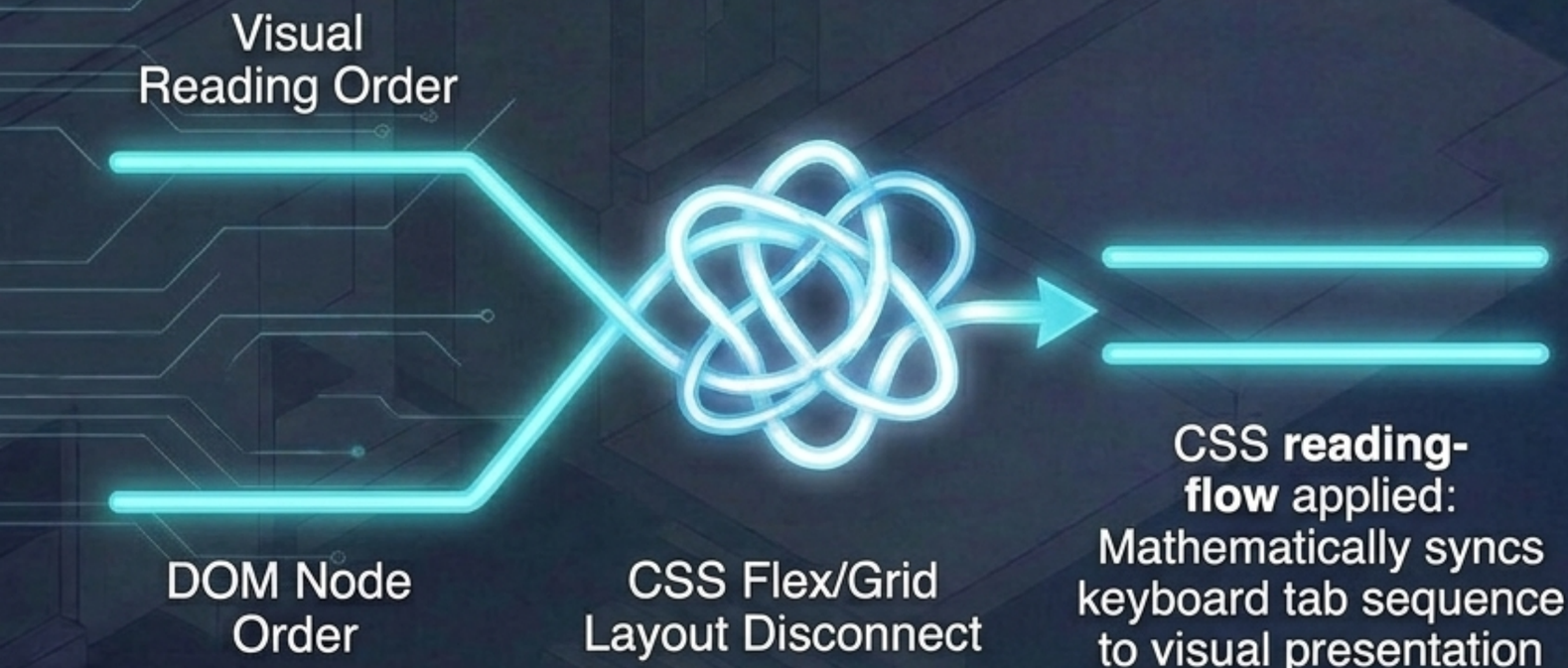
The Non-Jarring Upgrade: Ensure the <h1> has `outline: none` on focus. This avoids a massive, visually jarring focus ring appearing out of nowhere for mouse users, while successfully routing screen reader context to the top of the new view.

Gemini Notebook

6. Sticky Viewport Clearance, SPA & Shadow DOM

Bridging the Visual-vs-DOM Disconnect

Visual Reading Order vs. DOM Node Order



Shadow DOM & Focus:

Use delegatesFocus: true in the ShadowRoot config.



Clicking a custom element's empty space seamlessly routes focus to its first internal focusable element.

Gemini Notebook

7. Lab Exercise & Next Up

Engineering the Reusable Roving Hook

The Challenge: Implement a generic `useRovingTabIndex<T>` hook for Operations Desk toolbars.

`useRovingTabIndex<T>`

State Management

Manage a state array of `tabindex` values (0 and -1).

Event Interception

Attach `onKeyDown` listeners to explicitly intercept Arrow keys.

Ref Tracking

Export a `ref-updater` to track dynamically mapped elements.

Fallback Drill

Wire up the Delete Incident fallback sequence. Prove via testing that focus lands deterministically on the next row after deletion.

Gemini Notebook

7. Lab Exercise & Next Up

Locking In The Standard (AGENTS.md)

- [x] **Visibly Clear:** Dual-band indicators passing 3:1 contrast against all backgrounds. Focus is never obscured (WCAG 2.4.11).
- [x] **Programmatically Determinable:** Only one tabindex='0' per composite. Correct aria-activedescendant ID mappings.
- [x] **Trap-Free:** Escape safely exits deeply nested grid modes (WCAG 2.1.2).
- [x] **Logical Sequence:** Visual layout matches DOM/tab sequence natively or via CSS reading-flow.
- [x] **State Resilient:** restoreFocusOrFallback handles all unmounting trigger scenarios.

COMPLIANT // S10E10

AGENTS.md Updated: Focus is a lifecycle, not an afterthought.

Gemini Notebook

7. Lab Exercise & Next Up

Next Up: S10E11



Companion Lab: Accessible Charts and Data Visualisation

We leave the DOM and enter the Canvas/SVG.
Translating highly complex visual data into perceivable, navigable, and meaningful semantic structures.

Gemini Notebook