Engineer

ARIA

marcado extra que nombra controles para tecnología de apoyo

Qué es

ARIA es una especificación del W3C de atributos HTML que indica a las tecnologías de asistencia qué es un elemento de la página, cómo se llama y en qué estado está cuando el HTML por sí solo dice demasiado poco. Las letras significan Accessible Rich Internet Applications, y el título completo de la especificación, WAI-ARIA, añade Web Accessibility Initiative del W3C. La Recomendación vigente es la versión 1.2.

Un atributo `role` indica qué clase de elemento es: un componente como `button` o `tab`, una parte estructural como `list`, uno de los ocho roles de región de referencia, como `navigation` o `main`, o una región dinámica como `status`, cuyos cambios anuncia un lector de pantalla sin que se lo pidan. Los atributos `aria-*` se dividen después en propiedades, como `aria-labelledby`, y estados, como `aria-expanded`, que cambian según actúa el usuario. El navegador traslada todo ello a la API de accesibilidad del sistema operativo, que es lo que lee un lector de pantalla o un programa de control por voz.

ARIA cambia lo que se anuncia y nada más. Un `div` con `role="button"` se lee como un botón, pero no recibe foco e ignora Intro y Espacio hasta que un `tabindex` y un script aportan ambas cosas. De ahí lo que la guía Using ARIA del W3C llama la primera regla de uso de ARIA: cuando un elemento o atributo HTML nativo ya tiene la semántica y el comportamiento que necesita, úselo en lugar de dar otro propósito a un elemento y añadirle ARIA.

```
<!-- announced as a button, behaves like a div -->
<div role="button">Send</div>

<!-- announced, focusable and operable as a button -->
<button type="button">Send</button>
```

La otra mitad es el nombre accesible, las palabras que un lector de pantalla pronuncia para un control. Accessible Name and Description Computation 1.2 fija el orden: primero `aria-labelledby`, después `aria-label`, después la etiqueta nativa —un elemento `label`, un `alt`, un `title` de SVG—, a continuación el contenido de texto del elemento para los roles que toman de él su nombre y, al final, el atributo `title`. Por tanto, `aria-label` sustituye las palabras visibles de un botón en vez de añadirse a ellas.

Utilice el concepto en una decisión

Aplique «¿Qué es ARIA?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Interfaces que cargan rápido, funcionan en el teléfono y se comportan igual en los navegadores que utilizan sus clientes.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. Dirección del desarrollo integral de mapas GIS — cartografía integral, de la base de datos al navegador

Dirección del desarrollo integral de mapas GIS Póngase en contacto →

Por qué importa

La decisión que ARIA plantea a un equipo es construir o reutilizar. El criterio 4.1.2 de WCAG 2.2, Nombre, función, valor, exige que todo componente de interfaz exponga su nombre y rol al software, permita establecer programáticamente los estados y valores configurables por el usuario e informe cuando cambien. Un `button`, `select` o `details` nativo lo cumple de forma predeterminada. Una barra de pestañas, un combobox o una vista de árbol personalizados solo lo cumplen mediante ARIA y el script que mantiene veraces sus atributos ARIA.

Eso convierte un componente personalizado en un coste recurrente: manejo del teclado, gestión del foco, estados que sigan cada interacción y pruebas en más de un lector de pantalla, en cada rediseño. Juzgue una biblioteca de componentes por el patrón de ARIA Authoring Practices Guide del W3C que siga cada componente y por si el comportamiento de teclado de ese patrón sigue estando presente cuando se aparta el ratón.

La presión legal alcanza ahora a las empresas privadas. La Directiva de accesibilidad web, (UE) 2016/2102, cubre sitios y apps del sector público. European Accessibility Act, Directiva (UE) 2019/882, se aplica desde el 28 de junio de 2025 a una serie de productos y servicios de consumo, entre ellos el comercio electrónico y la banca de consumo, con una exención para las microempresas que prestan servicios. Ambas se apoyan en la norma armonizada EN 301 549, cuyos

requisitos web son WCAG de nivel AA. Nombre, función, valor se sitúa en el nivel A, por lo que un control cuyo nombre o rol permanezca oculto para un lector de pantalla falla en el nivel más bajo que comprueba una auditoría.

Cómo se aplica en la práctica

Las plantillas de este sitio usan ARIA con moderación. La barra superior de todas las páginas interiores es un `div` con `role="banner"`, en lugar del elemento `header` habitual. El comentario de la plantilla explica por qué: la hoja de estilo da a un `header` sin más calificadores el aspecto de la cabecera de altura completa de la página de inicio, por lo que ese elemento arrastraría su geometría a todas las páginas interiores; el rol aporta la región de referencia y nada más. Ese rol sitúa la marca, el enlace al inicio, dentro de una región de referencia. Es el caso limitado que permite la primera regla. El menú del sitio sigue el camino contrario: usa una pareja de `details` y `summary` que deja en manos del navegador el estado abierto, la semántica de botón y `aria-expanded`. El `nav` del menú y el `nav` del pie llevan valores distintos de `aria-label`, porque dos regiones de navegación con el mismo nombre incumplen la regla de unicidad del motor, y la comprobación de accesibilidad del sitio exige cero hallazgos. La implementación conservada del filtro del portafolio usa una región `role="status"` diseñada para permanecer en silencio hasta que el lector escriba. Ese filtro está actualmente desactivado con `searchEnabled = false`; ilustra la implementación conservada, no una interacción disponible en las páginas publicadas.

El relato de llevar el sitio a WCAG 2.2 nivel AA aporta la otra mitad de la historia de la cabecera: al activar las reglas de buenas prácticas del motor, una falló de inmediato porque, en todas las páginas interiores, la marca no pertenecía a ninguna región de referencia. La misma página enumera la ausencia de pruebas con tecnologías de asistencia reales entre los límites que publica.

El relato de los once defectos que el navegador daba por sanos muestra la distancia entre el marcado y la experiencia. Los once residían en cómo se renderizaba la página, no en lo que declaraba su marcado, y el marcado es todo lo que toca ARIA. Ocho enlaces del pie seguían siendo alcanzables con Tab después de que un estilo hubiera desactivado la interacción con el puntero sin cambiar nada más: el mismo patrón que `aria-hidden` sobre un elemento enfocable, donde el lector de pantalla pierde el elemento y el teclado lo conserva. Y solo estaban activas setenta de las ciento cinco reglas del motor, con la que señala una región de referencia ausente entre las desactivadas.

La última página muestra el mismo problema en una plataforma donde ARIA no interviene. Dar nombre a los diecisiete iconos de interfaz para lectores de pantalla describe una aplicación SwiftUI: sin HTML ni ningún atributo `aria-`. Su etiqueta de accesibilidad es la propia de SwiftUI, la contrapartida nativa de `aria-label`, no un caso de este atributo. Los controles dibujados con símbolos del sistema se leían por el nombre interno del símbolo, por lo que el botón de enviar se anunciaba como «arrow up circle, button», y diecisiete controles formados solo por iconos describían su dibujo en vez de su función. Cada uno recibió una etiqueta que nombra la acción y sigue el estado cuando eso importa, y ahora un linter exige una etiqueta a no más de ocho líneas de la construcción que crea un control de solo icono, de modo que un decimoctavo solo entra con etiqueta. La página declara el límite, que también vale para `aria-label`: la comprobación confirma que hay alguna etiqueta cerca, mientras que solo una persona que escucha puede juzgar si es la adecuada.

Dónde surgen los problemas

Un rol anuncia un componente al que nunca llega el teclado. Un `div` recibe `role="button"` y un manejador de clic. El lector de pantalla dice «botón», pero Tab lo omite e Intro no hace nada. La tercera regla de uso de ARIA exige que todo control ARIA interactivo funcione mediante teclado. Use el elemento nativo o construya todo el modelo de teclado del patrón APG y pruébelo con el ratón desconectado.

El foco llega a algo que el lector de pantalla trata como ausente. `aria-hidden="true"` está en un elemento enfocable o en un antecesor, como un panel lateral fuera de pantalla. Tab sigue llegando allí y el usuario oye silencio. La cuarta regla de uso de ARIA prohíbe esto. Retire también el contenido oculto del orden de tabulación con `hidden`, `display: none` o `inert`, y reserve `aria-hidden` para decoración que nunca reciba foco.

Se da un nombre a un elemento que no lo admite. Un `aria-label` en un `div`, `span` o `p` pretende explicarlo, pero WAI-ARIA 1.2 prohíbe dar nombre a los roles `generic` y `paragraph`, y los lectores de pantalla lo ignoran o lo leen de forma inconsistente. En un control, la etiqueta sustituye las palabras visibles y, cuando difieren, quien usa control por voz y dice lo que aparece en pantalla no acierta. Nombre elementos que puedan llevar un nombre y comience todo `aria-label` con las palabras visibles, como exige WCAG 2.5.3, Etiqueta en el nombre.

Dos roles se contradicen. Un rol explícito sustituye al propio del elemento. Un enlace con `role="button"` se anuncia como botón, pero desplaza la página al pulsar Espacio; `role="tab"` en un `h2` lo retira de la lista de encabezados; `role="navigation"` en un `ul` hace perder la lista y su número de elementos. La segunda regla de uso de ARIA pide cambiar la semántica nativa solo cuando sea necesario, y ARIA in HTML enumera los roles que puede aceptar cada elemento.

Una región dinámica no deja de hablar. `aria-live="assertive"` o `role="alert"` está en un recuento de resultados o una cuenta atrás que cambia con cada pulsación o segundo, y el lector de pantalla interrumpe al usuario para cada valor. Use actualizaciones de baja prioridad, que `role="status"` aporta por definición; anuncie una vez que se estabilice la entrada y mantenga la región en silencio al cargar.

Lecturas adicionales

- WAI-ARIA 1.2 — todos los roles, estados y propiedades, qué atributos admite cada rol y qué roles no pueden recibir un nombre.
- ARIA in HTML — los roles y atributos que puede llevar cada elemento HTML y cuáles son redundantes.
- Using ARIA — la guía práctica del W3C, con las reglas de uso de ARIA y un ejemplo de cada una.
- Accessible Name and Description Computation 1.2 — el algoritmo que decide qué nombre pronuncia un lector de pantalla para un control.
- Comprender Nombre, función, valor — qué exige el criterio 4.1.2 de WCAG 2.2 a cada componente, con técnicas y fallos.
- ARIA Authoring Practices Guide — el marcado y comportamiento de teclado de cada patrón habitual de componente, con ejemplos funcionales.

Glosario

Elija según sus restricciones reales

Describa el flujo de trabajo, las interfaces existentes y los usuarios que necesitan el cambio. Podemos definir una entrega comprobable y un traspaso que permita a su equipo mantener lo entregado.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es ARIA?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/aria/