Engineer

ARIA

extra märkning som ger en kontroll ett namn för hjälpmedel

Vad det är

ARIA, med fullständigt namn WAI-ARIA, är en W3C-specifikation för HTML-attribut som berättar för hjälpmedel vad ett element på sidan är, vad det heter och vilket tillstånd det befinner sig i, där HTML ensamt säger för lite. Bokstäverna står för Accessible Rich Internet Applications, och WAI i det långa namnet syftar på W3C:s Web Accessibility Initiative. Den gällande rekommendationen är version 1.2.

Attributet `role` anger vilken sorts sak ett element är: en widget som `button` eller `tab`, en strukturell del som `list`, en av de åtta rollerna för landmärken, som `navigation` eller `main`, eller en levande region som `status`, vars ändringar en skärmläsare läser upp utan att någon har bett om det. Attributen `aria-*` delas sedan i egenskaper, som `aria-labelledby`, och tillstånd, som `aria-expanded`, vilka ändras när användaren gör något. Webbläsaren översätter allt detta till operativsystemets tillgänglighets-API, och det är där en skärmläsare eller ett program för röststyrning hämtar sin information.

ARIA ändrar vad som läses upp och ingenting annat. En `div` med `role="button"` presenteras som en knapp, men den tar inte emot fokus och bryr sig varken om Enter eller mellanslag förrän ett `tabindex` och ett skript har lagt till båda delarna. Därav det som W3C:s guide Using ARIA kallar den första regeln för ARIA-användning: när ett inbyggt HTML-element eller attribut redan har den semantik och det beteende du behöver, använd det i stället för att ge ett annat element en ny uppgift och lägga på ARIA.

```
<!-- presenteras som en knapp, beter sig som en div -->
<div role="button">Send</div>

<!-- presenteras, tar emot fokus och manövreras som en knapp -->
```

```
<button type="button">Send</button>
```

Den andra halvan är det tillgängliga namnet, de ord en skärmläsare säger för en kontroll. Specifikationen Accessible Name and Description Computation 1.2 lägger fast ordningen: `aria-labelledby` först, sedan `aria-label`, därefter värdspråkets egen etikett (ett `label`-element, ett `alt` eller elementet `title` i SVG), sedan elementets textinnehåll för de roller som hämtar sitt namn därifrån och attributet `title` sist. En `aria-label` ersätter alltså de synliga orden på en knapp i stället för att komplettera dem.

Använd begreppet i ett beslut

Använd ”Vad är ARIA?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Gränssnitt som laddar snabbt, fungerar i mobilen och beter sig likadant i alla webbläsare dina kunder råkar använda.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Styrde full stack-utvecklingen av GIS-kartor — kartutveckling i full stack, från databas till webbläsare

Styrde full stack-utvecklingen av GIS-kartor Hör av er →

Varför det spelar roll

Valet som ARIA ställer ett team inför står mellan att bygga själv och att återanvända. Framgångskriteriet 4.1.2 i WCAG 2.2, Name, Role, Value, kräver att varje UI-komponent exponerar sitt namn och sin roll för programvara, att tillstånd och värden som användaren kan ställa in också går att ställa in programmatiskt, och att ändringar i dem meddelas. En inbyggd `button`, `select` eller `details` klarar det från start. En egenbyggd flikrad, kombinationsruta eller trädvy klarar det bara genom ARIA och det skript som håller ARIA-uppgifterna sanna.

Det gör en egenbyggd widget till en återkommande kostnad: tangentbordshantering, fokushantering, tillstånd som följer varje interaktion och tester i mer än en skärmläsare, vid varje omdesign. Bedöm därför ett komponentbibliotek efter vilket mönster ur W3C:s ARIA Authoring Practices Guide varje widget följer, och efter om mönstrets tangentbordsbeteende finns kvar när musen läggs undan.

Det rättsliga trycket når nu även privata företag. Direktivet om webbtillgänglighet, (EU) 2016/2102, omfattar den offentliga sektorns webbplatser och appar. European Accessibility Act, direktiv (EU) 2019/882, gäller från den 28 juni 2025 för en rad produkter och tjänster till konsumenter, bl.a. e-handel och banktjänster för konsumenter, medan mikroföretag som tillhandahåller tjänster är undantagna. Båda vilar på den harmoniserade standarden EN 301 549, vars krav på webben är WCAG på nivå AA. I Sverige genomförs direktiv (EU) 2019/882 genom lag (2023:254) om vissa produkters och tjänsters tillgänglighet, som trädde i kraft den 28 juni 2025, och kraven på digital service från offentliga aktörer står i lag (2018:1937) om tillgänglighet till digital offentlig service. Name, Role, Value ligger på nivå A, så en kontroll vars namn eller roll förblir dold för en skärmläsare underkänns redan på den lägsta nivå som en granskare kontrollerar.

Så ser det ut i praktiken

Mallarna på den här webbplatsen använder ARIA sparsamt. Den övre listen på varje undersida är en `div` med `role="banner"`, inte det vanliga elementet `header`. Kommentaren i mallen förklarar varför: stilmallen ger ett nakent `header` utseendet av startsidans hero-sektion i full höjd, så elementet skulle dra med sig den geometrin till varje undersida, medan rollen ger landmärket och ingenting annat. Det är rollen som placerar varumärkessymbolen, länken till startsidan, inuti ett landmärke, och det är just det smala fall som den första regeln tillåter. Webbplatsens meny går åt andra hållet: ett par av `details` och `summary`, som överlåter det öppna tillståndet, knappsemantiken och `aria-expanded` åt webbläsaren. Menyns `nav` och sidfotens `nav` har olika värden för `aria-label`, eftersom två navigeringslandmärken med samma namn bryter mot regelmotorns krav på unika namn, och webbplatsens tillgänglighetskontroll släpper bara igenom en ändring med noll fynd. Filterimplementationen för portfoliosidan som finns kvar i källkoden använder en region med `role="status"`, utformad för att vara tyst tills läsaren skriver. Filtret är för närvarande avaktiverat med `searchEnabled = false`; det visar den bevarade implementeringen, inte en interaktion som är tillgänglig på de publicerade sidorna.

Sidan om hur webbplatsen fördes till WCAG 2.2 nivå AA står för den andra halvan av historien om bannerrollen. Där körs regelmotorn för tillgänglighet med reglerna för bästa praxis påslagna, och en av de extra reglerna gav utslag i samma ögonblick som den slogs på: varje undersida hade varumärkessymbolen placerad utanför samtliga landmärken. Samma sida räknar bland sina publicerade begränsningar upp att ”det har inte gjorts några tester med riktiga hjälpmedel”.

Redogörelsen för elva brister som webbläsaren rapporterade som felfria visar hur långt det är mellan uppmärkning och upplevelse. Alla elva satt i hur sidan renderades och inte i vad uppmärkningen deklarerade, och uppmärkningen är det enda ARIA rör vid. Åtta länkar i sidfoten låg enligt sidan ”kvar i tabbordningen bakom en egenskap som tar bort pekarinteraktion och ingenting annat”. Det är samma mönster som `aria-hidden` på ett element som kan få fokus: skärmläsaren tappar elementet medan tangentbordet behåller det. Dessutom körde regelmotorn bara sjuttio regler av hundrafem, och den som skulle ha fångat det saknade landmärket hörde till dem som stod utanför uppsättningen.

Den sista sidan visar samma problem på en plattform där ARIA saknas helt. Alla sjutton ikoner i gränssnittet fick namn för skärmläsare i en SwiftUI-applikation, utan HTML och utan något attribut som börjar på `aria-`. Dess tillgänglighetsetikett är SwiftUI:s egen, den inbyggda motsvarigheten till `aria-label` och inte en variant av den. Kontroller ritade med systemsymboler lästes upp med symbolens interna namn, så skicka-knappen presenterades som ”arrow up circle, button”, och sjutton kontroller med enbart ikon beskrev sin egen grafik i stället för sin funktion. Var och en fick en etikett som namnger handlingen och följer tillståndet där det spelar roll, och en linter kräver nu en tillgänglighetsetikett inom åtta rader från den konstruktion som skapar en kontroll med enbart ikon, så ”en artonde kan inte läggas till utan en”. Sidan anger själv begränsningen, och den gäller `aria-label` lika mycket: regeln visar ”att det finns en etikett i närheten snarare än att etiketten är korrekt, och korrektheten kräver fortfarande att någon lyssnar på den”.

Var det går fel

En roll utlovar en widget som tangentbordet aldrig når. En `div` får `role="button"` och en klickhanterare. Skärmläsaren säger ”knapp”, men Tab hoppar förbi elementet och Enter gör ingenting. Den tredje regeln för ARIA-användning kräver att varje interaktiv ARIA-kontroll går att använda med tangentbordet. Använd det inbyggda elementet, eller bygg hela tangentbordsmodellen ur APG-mönstret och testa med musen urkopplad.

Fokus hamnar på något som skärmläsaren behandlar som frånvarande. `aria-hidden="true"` sitter på ett element som kan få fokus eller på en förfader till det, t.ex. en sidopanel som väntar utanför skärmen. Tab flyttar ändå dit, och användaren hör bara tystnad. Den fjärde regeln för ARIA-användning utesluter det. Ta bort dolt innehåll ur tabbordningen också, med `hidden`, `display: none` eller `inert`, och spara `aria-hidden` till dekoration som aldrig tar emot fokus.

Ett namn sätts på ett element som saknar plats för det. En `aria-label` på en `div`, `span` eller `p` ska förklara elementet, men WAI-ARIA 1.2 förbjuder namn på rollerna `generic` och `paragraph`, och skärmläsare hoppar över etiketten eller läser upp den olika från gång till gång. På en kontroll ersätter etiketten de synliga orden, och när de två skiljer sig åt missar en användare med röststyrning som säger det som står på skärmen. Ge namn åt element som kan bära ett, och börja varje `aria-label` med de synliga orden, som WCAG 2.5.3, Label in Name, kräver.

Två roller säger emot varandra. En uttrycklig roll ersätter elementets egen. En länk som får `role="button"` presenteras som knapp men rullar sidan när du trycker på mellanslag, `role="tab"` på en `h2` tar bort den ur listan över rubriker, och `role="navigation"` på en `ul` får listan och antalet punkter i den att försvinna. Den andra regeln för ARIA-användning säger att du bara ska ändra inbyggd semantik när du måste, och ARIA in HTML listar vilka roller varje element får ha.

En levande region pratar oavbrutet. `aria-live="assertive"` eller `role="alert"` sitter på ett antal träffar eller en nedräkning som ändras vid varje tangenttryckning eller sekund, och skärmläsaren avbryter användaren för varje nytt värde. Använd artiga uppdateringar, `polite` i specifikationens ord, som `role="status"` ger per definition, meddela först när inmatningen har stannat av och låt regionen vara tyst när sidan laddas.

Läs mer

- WAI-ARIA 1.2 — varje roll, tillstånd och egenskap, vilka attribut varje roll stöder och vilka roller som inte får namnges.
- ARIA in HTML — de roller och attribut som varje HTML-element får bära, och vilka av dem som är överflödiga.
- Using ARIA — W3C:s praktiska guide, med reglerna för ARIA-användning och ett exempel på var och en.
- Accessible Name and Description Computation 1.2 — algoritmen som avgör vad en skärmläsare säger att en kontroll heter.
- Understanding Name, Role, Value — vad framgångskriteriet 4.1.2 i WCAG 2.2 kräver av varje komponent, med tekniker och typiska fel.
- ARIA Authoring Practices Guide — uppmärkning och tangentbordsbeteende för varje vanligt widgetmönster, med fungerande exempel.

Ordlista

Välj utifrån era faktiska förutsättningar

Beskriv arbetsflödet, befintliga gränssnitt och användarna som behöver förändringen. Vi kan definiera en testbar leverans och en överlämning som gör att teamet kan underhålla lösningen.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är ARIA?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

welcome@engineer.company +45 50 10 61 84 Köpenhamn, Danmark

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/aria/