



Landstinget i Uppsala län – mellangranskning av app på Android



Fakta om granskningen

Beställare: Landstinget i Uppsala län

Hampus Andersson ,

hampus.andersson@regionuppsala.se

Utförd av: Melad Munther

Vår referens: Frida Sandberg

frida.sandberg@funka.com,

+46 76 319 15 64

Webbplatsens namn: UL

URL:

[Google](#)

[play v 3.5.1 \(10713\)](#)

Delar som inte ingått: Tredjepartsappar och externa sidor

Tidsperiod för undersökningen: 2021-05-16 till 2021-05-24

Utrustning:

- Samsung S20
- TalkBack

Bakgrund

Funka har genomfört en granskning för att bedöma hur väl gränssnittet uppfyller krav på tillgänglighet genom att kontrollera ett antal punkter. För varje punkt finns bedömningen:

- **Inga fel funna.** Gränssnittet uppfyller kravet i punkten tillfredsställande på de delar vi kontrollerat. Det kan innebära allt från mycket bra lösningar till acceptabla lösningar.
- **Inga fel funna men kan förbättras.** Gränssnittet uppfyller kravet i punkten tillfredsställande enligt gällande lagkrav. Att punkten är märkt med "kan förbättras" innebär att vi ändå ser förbättringsmöjligheter som inte inkluderas i lagkravet.
- **Underkänd.** Gränssnittet har klara brister vad gäller tillgänglighet som måste åtgärdas.
- **Inte aktuellt.** Gränssnittet använder inte den typen av lösning som kravet gäller, eller så är punkten av annan anledning inte bedömd inom ramen för granskningen.

Funkas metodik är utvecklad i nära samarbete med funktionshinderrörelsen och allt vi rekommenderar är testat i verkligheten. Vår verksamhet bygger på de internationella riktlinjerna för tillgänglighet, Web Content Accessibility Guidelines, WCAG. WCAG 2.1 AA är den nivå som för närvarande gäller för det svenska och europeiska regelverket.

Funkas långa erfarenhet av tillgänglighetsarbete och tester med användare med olika behov och förutsättningar, med och utan hjälpmedel, visar dock att varken EN-standarden eller WCAG räcker för att säkerställa tillgänglighet för alla användare. Vi har därför själva utarbetat testkriterier för punkter som kompletterar dessa standarder.

Funka har, som utnämnd Lead Translation Organisation, på uppdrag av W3C genomfört den auktoriserade översättningen av WCAG 2.0 till svenska och även fått i uppdrag att översätta WCAG 2.1. Denna översättning ligger också till grund för Vägledning för webbutveckling som är de officiella riktlinjerna för hur man bör arbeta med webbplatser inom offentlig sektor i Sverige.

Funkas specialister deltar sedan 2008 aktivt i arbetet med att ta fram EN301549 på uppdrag av EU-kommissionen. Vi är utnämnda experter i den Special Task Force som för ETSIs räkning harmoniserar EN-standarden med WCAG 2.1 och för att kunna uppfylla kraven i Webbtillgänglighetsdirektivet vad gäller dokument

och appar. Dessutom är vi invalda som tekniska experter i den svenska spegelkommittée hos Swedish Standards Institute som representerar Sverige i standardiseringsarbetet. Vi leder den expertgrupp som ger EUs medlemsländer och kommissionen stöd i övergångsperioden för Webbtillgänglighetsdirektivet och utformandet av tillsynsmetodik och tillgänglighetsutlåtande.

Standarder och standardiseringsorgan

[Tillgänglighetskrav vid upphandling: EN 301 549](#)

[Webbtillgänglighetsdirektivet \(på engelska\), öppnas i nytt fönster](#)

[Web Content Accessibility Guidelines 2.0, WCAG 2.0 \(på engelska\), öppnas i nytt fönster](#)

[Web Content Accessibility Guidelines 2.1, WCAG 2.1 \(på engelska\), öppnas i nytt fönster](#)

[Den auktoriserade svenska översättningen av WCAG 2.0, öppnas i nytt fönster](#)

[World Wide Web Consortium, W3C \(på engelska\), öppnas i nytt fönster](#)

[Web Accessibility Initiative, WAI \(på engelska\), öppnas i nytt fönster](#)

[ETSI \(på engelska\), öppnas i nytt fönster](#)

[CEN/CENELEC \(på engelska\), öppnas i nytt fönster](#)

Sammanfattning av granskningen

Appen UL på Android har mycket goda förutsättningar för att uppfylla kravet av en tillgänglig app. En stor fördel mot iOS versionen är att mycket av den tillgängliga tekniken redan finns och används i den här appen. UL på Android upplevs robust i dess funktioner och navigering men det finns enstaka brister i hur vissa funktioner fungerar för skärmläseanvändare.

Det finns två stora avsteg och det ena är att appen inte är speciellt anpassad för textförstoring. Vissa behållare flödar inte med vilket orsakar att text blir beskuren, samt att vissa fönster inte är anpassade för den större texten. Trots att texten går att förstora är den många gånger väldigt svår att uppfatta.

Det andra stora avsteget är att händelser i appen som visas visuellt inte läses upp för en skärmläseanvändare, vilket gör att de i flera fall kan gå miste om viktig information.

Appen är annars trevlig att använda med ett avskalat gränssnitt som också är fysiskt tillgängligt då många funktioner är nedflyttade på skärmen.

Prioriterade åtgärder

- Anpassa gränssnittet för större text.
- Säkra att alla visuella händelser och felmeddelanden också blir upplästa.
- Kontrollera att kontraster på text uppfyller kravet.
- Märk upp alla rubriker som rubriker.
- Möjliggör för skärmrotation.

Kommentarer och rekommendationer

Generellt

GE90: Sidor och funktioner beskrivs konsekvent i länkar, hänvisningar och rubriker

✓ Inga fel funna

Bakgrund

En viktig aspekt för att användarna ska uppleva sidorna som enkla och tydliga handlar om att webbplatsen är konsekvent i hur olika funktioner, länkar och områden beskrivs. Det skapar en osäkerhet hos användarna om webbplatsen använder "E-tjänster" på någon sida, men kallar samma tjänster för "Självservice" på andra sidor.

Kommentar

Vi bedömer att appen är konsekvent och inga fel funna.

Kopplade riktlinjer

[WCAG 2.0 - 3.2.4 \(AA\)](#)

[WCAG 2.1 - 3.2.4 \(AA\)](#)

[Vägledning för webbutveckling - R5 \(1\)](#)

[Vägledning för webbutveckling - R146 \(1\)](#)

[EN 301 549 - 9.3.2.4](#)

Html & css

HC10: Använd tekniker som går att använda på ett tillgängligt sätt

✓ Inga fel funna

Bakgrund

Det finns många olika tekniker som kan användas för att förmedla information och tjänster. Några exempel är html, css, java, Flash, Silverlight och pdf. En del av dessa fungerar bättre än andra ur ett tillgänglighetsperspektiv. Vilka tekniker som bör användas varierar också med tiden. För en del år sedan fungerade exempelvis Flash bra men idag är stödet för Flash inte så pass utbrett att tekniken bör användas längre.

Vi rekommenderar att information och funktioner som ska förmedlas på en webbsida i första hand skapas med html, css, wai-aria och JavaScript. Detta är tekniker som går att använda på ett fullt tillgängligt sätt samtidigt som stödet bland webbläsare och hjälpmedel är väl utbyggt.

Använd inte Java applets, Flash eller Silverlight. Det är tekniker som det visserligen finns vissa tillgänglighetsfunktioner för, men som i praktiken skapar svårigheter för användarna. Det kan i enstaka fall finnas skäl till att använda någon av de här teknikerna, men det ska vara i ett speciellt sammanhang där inget alternativ finns och där ni har full kontroll på användarna och deras utrustning.

Kommentar

Vi har sett inslag av tekniker som underlättar för användare. Utmärkande är Scalable pixels för textförstoring samt uppläsning av sökförslag och den återstående biljettiden.

Kopplade riktlinjer

[WCAG 2.0 - 4.1.1 \(A\)](#)

[WCAG 2.1 - 4.1.1 \(A\)](#)

[Vägledning för webbutveckling - R81 \(2\)](#)

[EN 301 549 - 9.4.1.1](#)

HC20: Tekniker används på ett tillgängligt sätt

✓ Inga fel funna

Bakgrund

Det räcker inte att välja en teknik som går att använda på ett tillgängligt sätt, du måste även använda tekniken korrekt. Detta handlar exempelvis om att använda riktiga html-element i varje sammanhang, se till att skriptbaserade funktioner fungerar med hjälpmedel och att användning av wai-aria inte skapar problem för hjälpmedelsanvändare.

Html

De flesta html-element har en semantisk betydelse, allt ifrån överordnade saker som att definiera navigationsområden och huvudinnehållsområden, till mer specifika saker som länkar och rubriker. Du ska använda element som ger ett mervärde till användaren, medan du bör undvika element som inte ger något mervärde.

Många av elementen som gör ett mervärde till användaren är så viktiga och/eller skapar så stora problem, att de har egna punkter senare i revisionen. Andra element är däremot väl inbyggda i de flesta publiceringsverktyg och/eller i fingrarna på de flesta utvecklare att vi väljer att inte lyfta dem i egna punkter. Det sistnämnda gäller exempelvis att koda länkar med <a>-element.

Grundregeln är enkel, håll det enkelt, använd element till det som de är avsedda och försök hålla strukturen så enkel som möjligt.

Skript

Skript ställer större krav på användarnas hjälpmedel än vad enbart html och css gör. Det innebär att risken för problem ökar. Vi rekommenderar därför att alla komplexa funktioner testas med de vanligaste hjälpmedlen för att säkra att inga onödiga svårigheter uppstår.

Exempel på problem som är vanliga är att användare med hjälpmedel inte får fokus på lager som öppnas dynamiskt eller inte kommer åt sökförslagen i sökfunktionen. Detta beror på att nya lager och ny information ofta läses in sist i sidans struktur och inte vid det objekt där användaren har fokus. Ett annat vanligt problem är att klickbara ytor skapas med skript i stället för med länkar och knappar. Då uppfattar inte hjälpmedlen detta som riktiga länkar/knappar och presenterar det därför inte som interaktiva objekt.

WAI-ARIA

Wai-aria är en specifikation av kod som gör det möjligt att ge information om webbplatsen till hjälpmedelsanvändare. Med wai-aria kan man exempelvis märka upp dynamiskt innehåll så att hjälpmedlen har lättare att hantera och presentera informationen till användarna.

Vi rekommenderar att wai-aria används, men det måste göras med försiktighet. Använd inte wai-aria för att rätta dålig html. Det är exempelvis ingen bra idé att använda `role="button"` på ett `div`-element i stället för att använda en riktig knapp. Använd inte heller wai-aria utan att ha kontroll på hur det faktiskt fungerar för användarna, ett vanligt problem är att man lägger in all wai-aria man kan tänka sig med följden att hjälpmedlen blir extremt pratiga helt i onödan.

Stödet för wai-aria varierar mellan olika versioner av hjälpmedel och webbläsare. Det gör att vi tills vidare rekommenderar att testa alla dynamiska lösningar praktiskt med olika hjälpmedel, enheter och versioner.

[Information om wai-aria \(W3C:s webbplats\).](#)

Vilka webbläsare och hjälpmedel ska tekniken fungera för?

Grundprincipen är att använda fördelarna i senare tekniker för att öka användarnytan. Samtidigt vet vi att många användare som har lite större svårigheter gärna undviker att uppdatera operativsystem, webbläsare och hjälpmedel. Exakt vad systemet ska stödja varierar från fall till fall men utgå ifrån att det finns användare som sitter med äldre utrustning och försök så långt rimligt stödja äldre versioner av webbläsare och hjälpmedel. Det betyder inte att det måste fungera perfekt i äldre versioner, men så långt möjligt bör det gå att hantera gränssnittet även med äldre versioner.

Testa med hjälpmedel

För att vara säker på att skript inte orsakar problem för hjälpmedelsanvändare behöver detta oftast testas rent praktiskt. Exakt vilka hjälpmedel som bör testas varierar beroende på lösning. Det är inte heller bara att sätta sig ned med ett hjälpmedel och köra igenom tjänsten. Det krävs kunskap om hur användarna fungerar, hur de använder hjälpmedlen, hur det uppfattas om man exempelvis inte ser och så vidare. Som exempelvis utvecklare är det oerhört svårt att testa lösningen objektivt eftersom du har en förkunskap som användarna saknar. Vi rekommenderar därför att ni låter experter testa detta.

Kommentar

De tekniker som används upplever vi fungerar bra, punkten bedöms som godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 4.1.1 \(A\)](#)

[WCAG 2.1 - 4.1.1 \(A\)](#)

[WCAG 2.1 - 4.1.2 \(A\)](#)

[WCAG 2.0 - 4.1.2 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.4.1.1](#)

[EN 301 549 - 9.4.1.2](#)

HC30: Html-koden innehåller inga allvarliga fel

– Inte aktuellt

Bakgrund

Det finns ett antal olika html-standarder, exempelvis html5 och xhtml 1.0 strict. Varje standard har sina regeldokument för vilka element och attribut som får användas och på vilket sätt. När du kodar html måste du först ta ställning till vilken standard koden ska utgå ifrån. Vi rekommenderar att i första hand överväga html5 eftersom det här finns en del attribut och element som ökar tillgängligheten jämfört med äldre standarder. Det bygger dock på att html5 används försiktigt. Det finns exempelvis problem med en del attribut och element där det dels inte finns stöd i alla webbläsare och dels inte finns ett homogent stöd bland olika hjälpmedel.

En viktig princip är också att effektivisera koden, det handlar bland annat om att bryta ut skript och stilmallar till egna filer och att effektivisera strukturen och formateringen.

!DOCTYPE

För att webbläsare, hjälpmedel och andra verktyg ska tolka koden korrekt måste de veta vilken standard den aktuella sidan är kodad i. Detta måste därför anges överst i koden på webbplatsens samtliga html-sidor med en !DOCTYPE-deklaration.

!DOCTYPE deklareras enligt följande i html5:

```
<!DOCTYPE html>
```

Om ni väljer att följa xhtml i stället för html5 ska deklarationen innehålla hänvisningar till regeldokument för standarden också, så här ser en deklaration av xhtml 1.0 transitional ut:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-  
transitional.dtd">
```

Dtd-filen definierar exakt vilka element, attribut och olika värden som är tillåtna enligt standarden.

Html5

Om du väljer att följa html5 så ska du göra det med försiktighet eftersom stödet för olika delar i standarden varierar mellan webbläsare och hjälpmedel. Ta in nya delar i html5 standarden när det ökar användarnyttan och det inte försvårar för användare.

Just nu bör du undvika html-elementen section och article om det inte finns särskilda skäl att använda dessa element. Anledningen är att dessa skapar en struktur som kan påverka sidornas rubrikstruktur. Olika hjälpmedel tolkar detta på olika sätt vilket gör att användare som är beroende av hjälpmedlen för att förstå sidornas struktur (exempelvis gravt synskadade användare) kan få olika rubrikstrukturer beroende på version på hjälpmedlet.

Tänk också på bakåtkompatibiliteten, om en något äldre webbläsare inte har stöd för html5 tekniken så ska det finnas en fallbacklösning. Läs mer om detta på T1a ovan.

Följ vald standard

Det räcker inte att med !DOCTYPE deklarerar att en webbsida ska följa en viss standard. Det är även viktigt att faktiskt följa den standard som anges i koden. Standarden är framtagen för att alla utvecklare av webbplatser, webbläsare och hjälpmedel ska ha en gemensam källa att titta på för att säkra att innehållet på webbplatsen presenteras på ett optimalt sätt för besökaren.

Om webbplatsen inte följer standarden riskerar ni att användaren får problem att ta del av innehållet. Blir problemen alltför stora riskerar ni att helt stänga ute grupper av användare.

Det finns ett flertal olika valideringsverktyg för att kontrollera en webbplats kod. W3C har en egen html-validator som kan nås på adressen:

<http://validator.w3.org/>

Alla avsteg från vald standard resulterar inte i lika stora tillgänglighetsproblem. En del avsteg är allvarigare än andra. W3C pekar i wcag 2.1 ut fyra principer som koden som ett minimum måste följa:

- Använd kompletta start och avslutningstaggar.
- Ange inte samma id-värde på två olika element på samma sida.
- Ange inte samma attribut två gånger i samma element (exempelvis två alt-attribut i samma img-element).
- Nästla elementen enligt standarden.

Dessa betraktas som allvarliga och om de förekommer kan inte gränssnittet sägas uppfylla kraven i wcag 2.1.

Kommentar

För validera kod på Android kan man använda Lint som är en funktion i Android studio. Det kan kontrollera att koden upprätthåller så god standard som möjligt. För att kunna testa måste man ha tillgång till källkoden, vilket inte är åtkomligt på appar. Punkten bedöms som ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 4.1.1 \(A\)](#)

[WCAG 2.1 - 4.1.1 \(A\)](#)

[Vägledning för webbutveckling - R80 \(1\)](#)

[Vägledning för webbutveckling - R81 \(2\)](#)

[Vägledning för webbutveckling - R84 \(1\)](#)

[EN 301 549 - 9.4.1.1](#)

HC80: Innehållets läsordning är logisk

✓ Inga fel funna

Bakgrund

När du styr layout och presentation med hjälp av css kan du positionera olika områden visuellt helt oberoende av exakt var de är placerade i html-koden. Hjälpmedel läser dock webbplatsens html-kod uppifrån och ned vilket innebär att presentationsordningen blir den ordning innehållet är lagt i koden och inte den visuella ordning som css-koden bestämmer. Innehållets ordning måste därför vara logisk och spegla den visuella ordningen.

Kommentar

Innehållets läsordning följer en logisk ordning, inga fel återfinns.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.2 \(A\)](#)

[WCAG 2.1 - 1.3.2 \(A\)](#)

[Vägledning för webbutveckling - R92 \(4\)](#)

[EN 301 549 - 9.1.3.2](#)

Gränssnittets flexibilitet

RD10: Gränssnittet fungerar väl i olika skärmbredder

– Inte aktuellt

Bakgrund

För de flesta webbgränssnitt är det rimligt att tänka sig att användaren ibland kommer att använda en liten och smal skärm och ibland en bredare. Då är det också rimligt att gränssnittet fungerar väl med olika skärmbredder.

Oftast är det vettigast att dela upp konceptet i tre huvudkategorier:

- Stora skärmar, vanliga desktopdatorer och bärbara datorer med normalstora skärmar.
- Mellanstora skärmar, exempelvis vanliga surfplattor eller bärbara datorer med små skärmar.
- Små skärmar, exempelvis mobiltelefoner.

Det kan finnas behov av en finare indelning beroende på gränssnittet, men det är viktigt att det fungerar väl i hela spektrat från små mobiler till stora desktopdatorer och att det fungerar både i stående och liggande visning.

Viktiga principer

Det finns olika principer och metoder som kan användas för att bygga responsiva lösningar. Om radlängder och ytor anpassar sig efter fönsterbredden är det bra, men det viktigaste är att gränssnittet är möjligt att använda i allt från smala skärmar till mycket breda skärmar och att presentationsordningen är logisk oavsett skärmbredd. Det innebär att hindra att textrader blir för korta så att de inte klarar av att visa hela ord, eller för långa. Textraderna ska inte visa mer än maximalt 70 tecken per rad inklusive mellanslag.

Så fort menyn kollapsar till en menyknapp ökar problemen för användaren. Därför är det viktigt att försöka behålla en meny där nivå ett är direkt synlig så långt som

möjligt. Det innebär exempelvis att menyens nivå ett ska vara synlig i mellanstorleken så långt möjligt.

I första hand bör det vanliga gränssnittet fungera oberoende av skärmbredd, men i specifika situationer kan det vara önskvärt att erbjuda ett alternativt, förenklat gränssnitt för små skärmar. Då är det viktigt att användarna inte tvingas till en viss version utan att de kan välja det ordinarie gränssnittet även på små skärmar.

Detta löses genom att erbjuda en länk till det ordinarie gränssnittet från mobilversionen. Det kan också vara bra att erbjuda en länk till mobilversionen från det ordinarie gränssnittet trots att användaren sitter vid en stationär dator.

Exempelvis uppskattar en del användare med koncentrationssvårigheter den förenklade visningen i mobilversionen.

Kommentar

För denna analys hade vi ej möjlighet att testa på en Android-surfplatta. Punkten bedöms som ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 1.4.8 \(AAA\)](#)

[WCAG 2.1 - 1.4.8 \(AAA\)](#)

[WCAG 2.1 - 1.4.10 \(AA\)](#)

[Vägledning för webbutveckling - R39 \(2\)](#)

[Vägledning för webbutveckling - R91 \(2\)](#)

[EN 301 549 - 9.1.4.10](#)

RD30: Gränssnittet ska kunna visas och användas utan att användaren behöver skrolla i mer än en riktning

✓ Inga fel funna

Bakgrund

Det är omöjligt att förutse hur en användare väljer att visa eller att använda ett gränssnitt. Det innebär att gränssnittet måste vara flexibelt och anpassa sig till den skärmbredd som användaren väljer att visa det i. Gränssnittet ska vara responsivt och anpassa sig till användarens skärmbredd utan att innehåll eller funktioner går förlorade. Om innehållet eller funktioner inte ryms på användarens skärm är skrollning endast tillåten i höjdlängd eller sidlängd, men inte i båda riktningarna samtidigt. Gränssnittet måste fungera med denna förutsättning från

minst 320 pixlars bredd eller 256 pixlars höjd. Ibland kan undantag vara nödvändiga för särskilt innehåll, exempelvis tabeller, diagram eller illustrationer.

Kommentar

Vi har inte upplevt att användaren behöver skrolla i två olika led samtidigt, därför bedöms punkten som inga fel funna.

Kopplade riktlinjer

WCAG 2.1 - 1.4.10 (AA)

EN 301 549 - 9.1.4.10

RD40: Webbplatsen är fullt användbar och läsbar vid förstoring upp till och med 200 procent

× Underkänd

Bakgrund

En del webbläsare, exempelvis Internet Explorer, ger användaren möjlighet att anpassa textstorleken om texterna lagts in med relativa mått (exempelvis em och %). Denna inställning har idag hamnat i skuggan av den mer utspridda zoom-funktionen som förstorar allt, oavsett måttenheter på texten.

Det är inte längre ett krav att bygga webbplatsen med relativa mått, men det är viktigt att texten inte blir svår att läsa och funktionerna svåra att använda om användaren ändrat inställningarna för textstorlek i webbläsaren. Om webbplatsen byggs med relativa storlekar måste det alltså göras på ett bra sätt som inte förvränger visningen av sidorna i webbläsaren. Ett enkelt sätt att testa detta är att använda funktionen Textstorlek i Internet Explorer. Ta fram menyn (syns den inte så tryck på Alt-tangenten) och välj "Visa" > "Textstorlek". Testa både "Störst" och "Minst".

Det måste också vara möjligt för användaren att zooma webbplatsen upp till 200 % utan att ytor lägger sig över varandra och gör texten svårläst eller funktionerna svåra att använda. Detta gäller inte enbart stora skärmar med HD-upplösning utan även på mindre skärmar och i mobiltelefoner.

Tänk också på att responsivt byggda gränssnitt ska anpassa sig när användaren zoomar. Det ska alltså även med zoom i ett responsivt gränssnitt gå att komma åt alla funktioner och all information. Det finns givetvis en gräns där en allt för liten skärm med allt för stor zoom inte fungerar men när användaren når dit ska denne fortfarande kunna zooma vidare även om det då kan innebära att användaren måste skrolla i sidled för att kunna läsa all text.

Du säkerställer att användaren kan zooma även i mobila enheter med följande kod i sidhuvudet:

```
<meta name="viewport" content="user-scalable=yes">
```

Attributet content kan innehålla många fler delattribut och tillhörande värden för att styra presentationen på olika skärmar. Användning av delattributet maximum-scale kan hindra användarens möjlighet att zooma även om du angett user-scalable=yes. Vi rekommenderar därför att du inte använder maximum-scale. Övriga delattribut har vi så här långt inte sett några problem med.

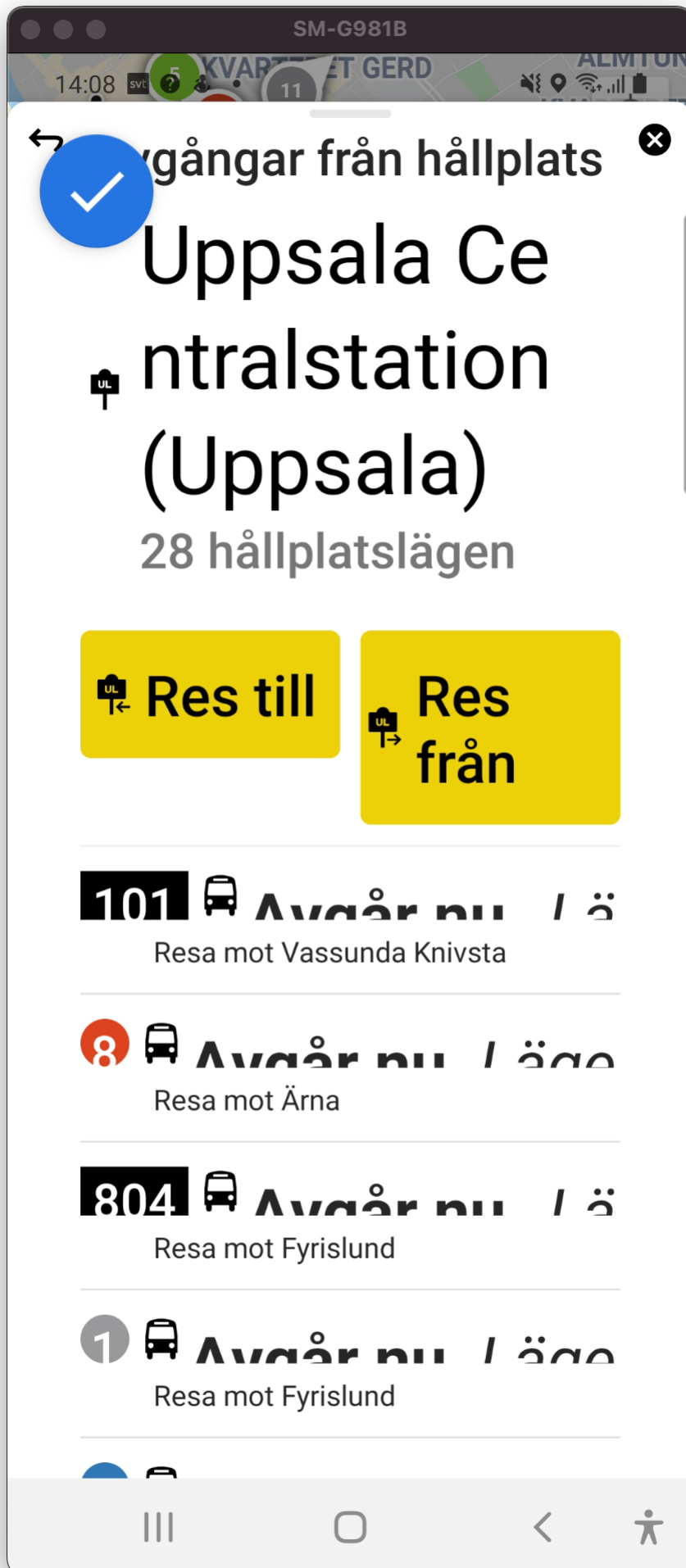
Den exakta lösningen för hur förstoring och ändring av fönsterstorlek ska hanteras måste skraddarsys från fall till fall, men här är några korta rekommendationer för vanliga webbgränssnitt:

- Webbplatsen ska anpassa sig automatiskt efter fönsterstorleken, men det är viktigt att lägga in mekanismer som hindrar att radlängderna överstiger 70 tecken inklusive mellanslag eller blir så korta att inte hela ord kan visas.
- Om användaren ändrar textförstoringsfunktionen i webbläsaren ska det inte innebära att innehåll inte går att läsa eller funktioner inte går att använda.
- Förstoring via webbläsare med zoomfunktion ska minst fungera upp till 200 %.
- Vid förminskning av fönster eller textstorlek ska texten vara fortsatt möjlig att läsa och funktionerna möjliga att använda.

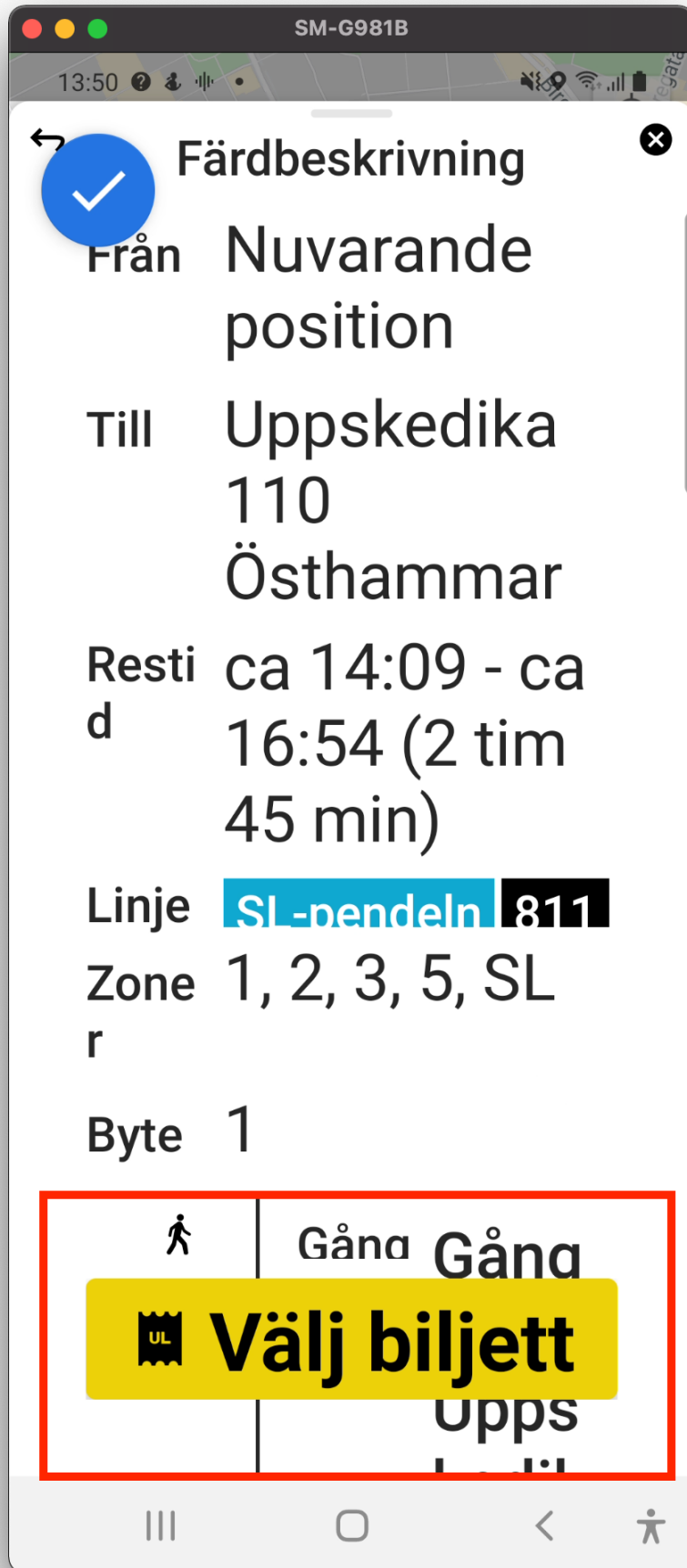
Kommentar

Vi har kunnat öka texten till 200%, vilket är positivt, men det förde med sig problem där gränssnittet inte var anpassat för den större texten. Detta lagkrav kan medföra att text blir avklippt på vissa ställen då appar har begränsat med utrymme att jobba på. I vanliga fall brukar vi förbise sådana problem, men i detta fall är texten direkt oläslig då gränssnittet inte flödar med som det skall.

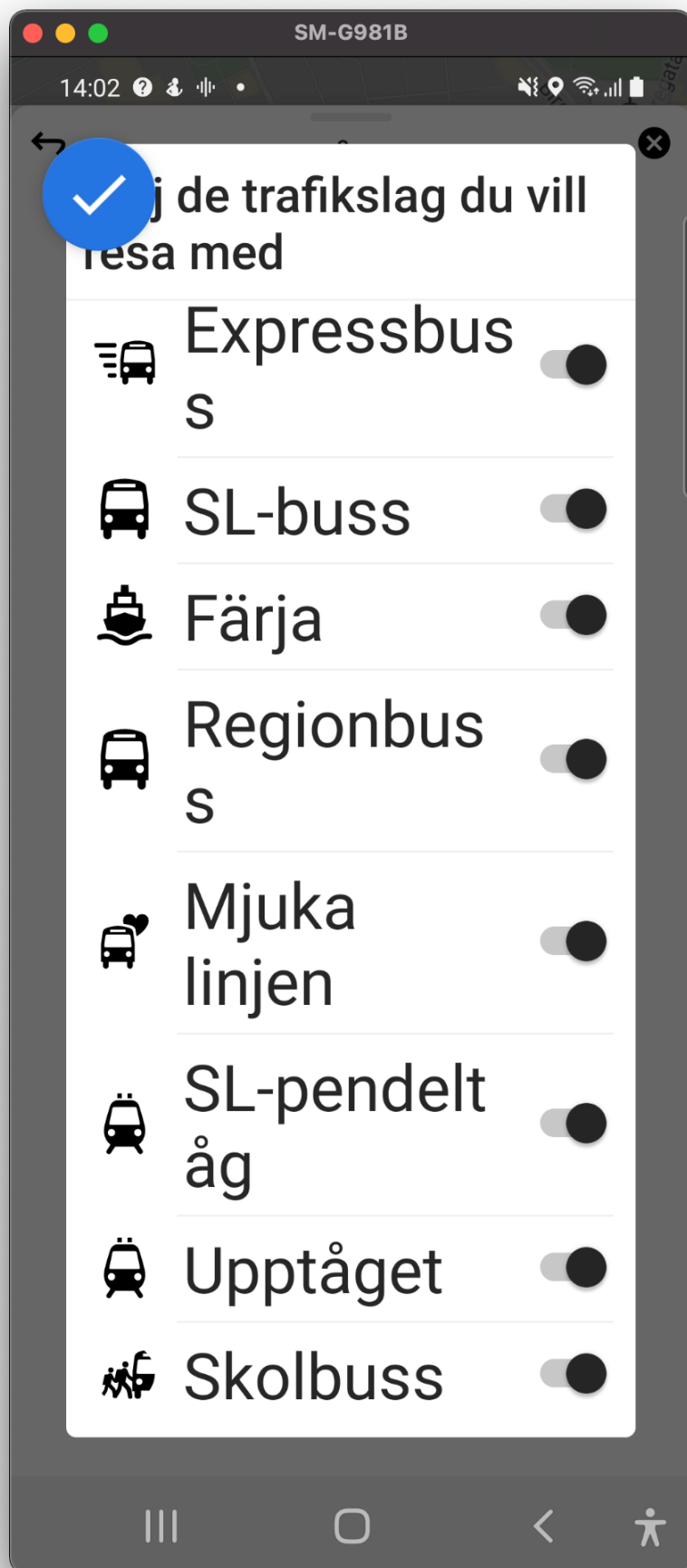
Det är vanligt att text inte får plats i sin behållare.



Ett annat problem fann vi i färdbeskrivningen där endast innehållet bakom den röda rektangeln gick att scrolla. Den lilla ytan och placeringen av biljettknappen gjorde det väldigt svårt att se texten.



När man öppnade filtervyn för trafikslag blev man fast där då knapparna "Avbryt" och "OK" ej var synliga och det gick ej att scrolla ned på sidan. Det gick inte heller att backa med telefonens knappar.



Rekommendationer

- Se över så att behållare också följer med när texten förstoras.
- Se över möjligheten att flöda om reglage/element så att de får plats när texten förstoras.
- Undersök om huvudinnehållet går att bredda. Just nu nyttjar det en begränsad bredd av telefonens skrämyta.

Kopplade riktlinjer

WCAG 2.0 - 1.4.4 (AA)

WCAG 2.0 - 1.4.8 (AAA)

WCAG 2.1 - 1.4.4 (AA)

WCAG 2.1 - 1.4.8 (AAA)

Vägledning för webbutveckling - R34 (1)

EN 301 549 - 9.1.4.4

RD60: Användarna kan nå all information och använda alla funktioner oberoende av skärmentoriering

× Underkänd

Bakgrund

När innehåll presenteras för en användare ska det inte vara låst till en liggande eller stående vy. Användaren ska kunna välja hur innehållet visas. Ibland är det inte möjligt utan att syftet med funktionen går förlorad. Film och tabeller ska vara responsiva och kunna visas oavsett orientering, om det inte finns särskilda skäl. Film i fullskärmsläge kräver ofta att enheten har en liggande orientering, men ska kunna spelas upp även i stående orientering.

I olika situationer kan användaren behöva använda mobiltelefoner eller surfplattor i en särskild orientering, exempelvis om enheten fästs i en hållare på en rullstol. Gränssnittets orientering ska då anpassas efter användarens förutsättningar. En konsekvens blir att upplevelsen inte alltid blir optimal, men det ska fungera. Ett exempel på detta är filmvisning i stående vy.

Denna riktlinje gäller inte förändringar av innehåll eller funktioner på grund av bildskärmens storlek, utan gäller enbart skärmentoriering.

Kommentar

Innehållet i telefonen är låst till stående läge och punkten blir därför underkänd.

Rekommendationer

- Möjliggör för skärmrotation.

Kopplade riktlinjer

WCAG 2.1 - 1.3.4 (AA)

EN 301 549 - 9.1.3.4

Navigation & länkar

NL20: Gränssnittet kan styras med enbart tangentbordet både på mobil och stor skärm

✓ Inga fel funna

Bakgrund

Det finns en mängd olika sätt att styra en dator eller telefon på. Det finns olika lösningar för att flytta muspekaren, klicka och mata in text. Genom att göra gränssnittet möjlig att styra med mus, tangentbord och pekskärm har användaren möjlighet att själv välja styrlösning. För en del användare finns inga möjliga alternativ. En del användare kan inte använda ett vanligt tangentbord utan måste navigera med musen. Andra användare kan inte använda någon form av muslösning utan måste navigera med tangentbordet.

En grundprincip i wcag är att det ska gå att styra gränssnittet enbart med tangentbord, oavsett om det är på en stor eller liten skärm. För att kunna anses följa riktlinjerna måste det fungera.

Undantag från denna punkt kan göras om det exempelvis rör en funktion som inte på något rimligt sätt skulle gå att göra möjlig att använda oberoende av inmatningsenhet.

Vanliga problem

Undvik att lägga in funktioner som kräver att användaren klickar eller håller muspekaren över olika områden, exempelvis menyer som kräver att användaren hovrar med musen. Undvik också listor som uppdaterar sidan eller flyttar fokus så fort värdet i listan ändras. De här typerna av lösningar skapar stora problem framförallt för personer som måste navigera med tangentbordet, men även för personer som har nedsatt precisionsförmåga eller som sitter i oroliga miljöer.

Var också försiktig med att ändra interaktionsmönster som användaren är van vid. Exempelvis är användare som navigerar med tangentbordet vana vid att använda tabbtangenten för att flytta mellan länkar och formulärsobjekt, om då plötsligt

användaren förväntas använda piltangenterna i stället kommer många användare att få problem.

Tänk också på att kombinationen med responsiv design och zoom inte sällan resulterar i en mobilvisning även för användare på desktop. Det är därför viktigt att det fungerar oberoende av inmatningsenhet även i mobilvisning.

Kommentar

Vi upplever att användandet med ett externt tangentbord är smidigt och funktioner är åtkomliga. Punkten godkänns.

Kopplade riktlinjer

[WCAG 2.0 - 2.1.1 \(A\)](#)

[WCAG 2.0 - 2.1.2 \(A\)](#)

[WCAG 2.0 - 2.1.3 \(AAA\)](#)

[WCAG 2.0 - 3.2.1 \(A\)](#)

[WCAG 2.0 - 3.2.2 \(A\)](#)

[WCAG 2.1 - 2.1.1 \(A\)](#)

[WCAG 2.1 - 2.1.2 \(A\)](#)

[WCAG 2.1 - 2.1.3 \(AAA\)](#)

[WCAG 2.1 - 3.2.1 \(A\)](#)

[WCAG 2.1 - 3.2.2 \(A\)](#)

[Vägledning för webbutveckling - R129 \(1\)](#)

[Vägledning för webbutveckling - R130 \(1\)](#)

[Vägledning för webbutveckling - R143 \(1\)](#)

[Vägledning för webbutveckling - R144 \(1\)](#)

[EN 301 549 - 9.2.1.1](#)

[EN 301 549 - 9.2.1.2](#)

[EN 301 549 - 9.3.2.1](#)

[EN 301 549 - 9.3.2.2](#)

NL30: Tangentbordsnavigationen följer en logisk ordning

! Inga fel funna men kan förbättras

Bakgrund

När användare navigerar med tangentbordet kan denne inte styra ordningen på samma sätt som en användare som navigerar med mus eller pekskärm. I stället är det webbplatsen som styr navigationsordningen. Ofta handlar det här enbart om tabbordningen. Det vanligaste sättet att hoppa mellan objekt med tangentbordet är att trycka upprepade gånger på tabbtangenten, då flyttas fokus mellan webbplatsens olika länkar och formulärobject. Tabbordningen är normalt sett den ordning i vilket innehållet lagts i koden. Varje gång användaren trycker på tabbtangenten hoppar denne från en länk eller formulärobject till nästa.

Attributet `tabindex` kan användas för att styra tabbordningen. Det kan i enstaka fall finnas skäl att använda detta attribut för att exempelvis hjälpa till att hindra ett objekt från att få fokus, men alla objekt som användaren ska tabba till bör ligga i en logisk ordning i sidans struktur och då behövs inte `tabindex` för att styra ordningen.

`Tabindex` fungerar enligt följande:

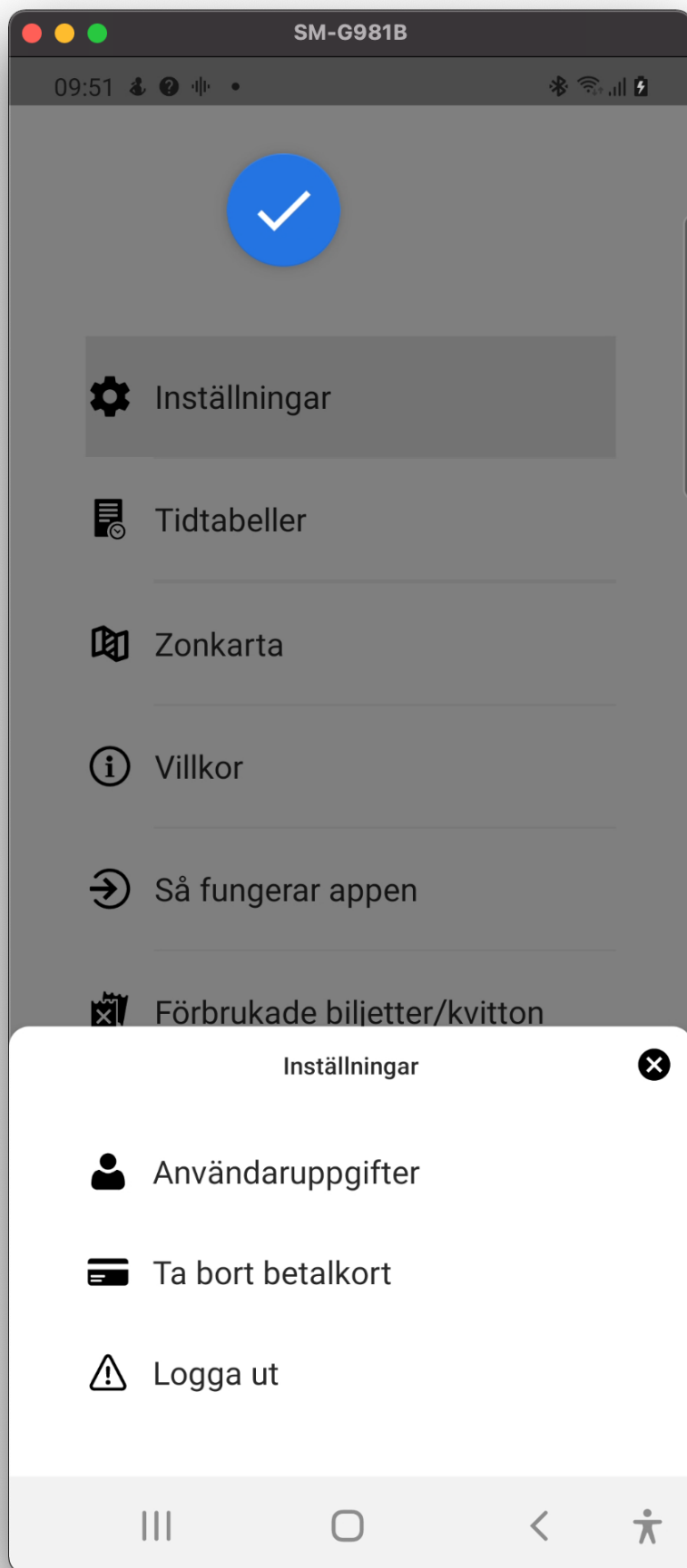
- **`Tabindex = 0`** lägger till inaktiva objekt, exempelvis en `div`, i tabbordningen på det ställe de naturligt ligger i sidans struktur. Vår rekommendation är att i första hand använda `html`-element som i sig är interaktiva (exempelvis länkar och knappar), men i enstaka situationer kan det finnas behov av att låta användaren tabba till andra objekt och då ska `tabindex=0` användas.
- **`Tabindex > 1`** prioriterar objekten och ändrar navigationsordningen så att den inte följer sidans struktur. Ordningen blir från lägst nummer och uppåt. Efter att användaren kommit till objektet med högsta `tabindex` angivelsen så fortsätter tabbordningen från detta objekt i koden. Det gör att användaren inte kommer åt objekt som ligger tidigare i sidans struktur om de inte har `tabindex` angivet.
- **`Tabindex < 0`** gör att objektet inte går att tabba till, men att det kan få fokus genom skript. Används exempelvis för att med skript sätta fokus på ett felmeddelande så att det förmedlas till hjälpmedelsanvändare. Normalt sett används värdet `-1` för detta.

I en del funktioner navigerar användaren med andra tangenter också, exempelvis piltangenten för att flytta fokus mellan olika radioknappar. Det är viktigt att inte

enbart tabbordningen är logisk utan att även övrig tangentbordsnavigation har en logisk ordningsföljd.

Kommentar

Tangentbordsnavigeringen är logisk och fungerar väl. Vår anmärkning ligger i att fokus hamnar utanför modalfönster och det är möjligt att klicka på och byta vy. Fokus skall stanna inuti det aktiva fönstret.



Rekommendationer

- Säkerställ att fokus stannar i modalfönstret när ett externt tangentbord används.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.3 \(A\)](#)

[WCAG 2.1 - 2.4.3 \(A\)](#)

[Vägledning för webbutveckling - R136 \(1\)](#)

[EN 301 549 - 9.2.4.3](#)

NL40: Fokus visas visuellt tydligt när användaren navigerar med tangentbordet

– Inte aktuellt

Bakgrund

När användaren förflyttar sig med tangentbordet på en webbsida eller i en funktion måste det visuellt tydligt framgå var fokus är. När man exempelvis tabbar sig genom menyn kan det menyalternativ som just nu har fokus framhävas med en ram eller genom att färgerna inverteras. I en del funktioner kan man förflytta sig med piltangenterna och då gäller samma rekommendation: Det objekt som har fokus ska vara visuellt tydligt framhåvet.

Även dolda länkar och genvägar ska lyftas fram och visas visuellt när de får fokus vid tangentbordsnavigering.

Markeringen av fokus ska vara synlig. Det kan till exempel vara en något tjockare ram, en tjockare understrykning, invertering eller en bakgrundsplatta som dyker upp.

Kommentar

Fokusmarkeringen sköts av operativsystemet och punkten bedöms därför som ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.7 \(AA\)](#)

[WCAG 2.1 - 2.4.7 \(AA\)](#)

[Vägledning för webbutveckling - R34 \(1\)](#)

[EN 301 549 - 9.2.4.7](#)

NL50: Det finns genvägar för att underlätta navigering med tangentbord

– Inte aktuellt

Bakgrund

För användare som navigerar med tangentbordet kan det vara besvärligt att behöva tabba sig igenom hela menystrukturen på alla sidor. Har man väl hittat den sida man sökte är det innehållet man är intresserad av.

För att underlätta för dessa användare ska det finnas genvägar på sidorna. Den vanligaste genvägen är förbi navigationen till sidans innehåll, men i olika sammanhang kan det också vara lämpligt med länkar tillbaka till sidans topp eller till olika menygrupper på sidan.

Denna typ av genvägar skapas genom att lägga in länkar som leder till element längre ner på sidan. Länkar direkt till innehållet förbi navigationen ska vara visuellt dolda men visas när användaren tabbar sig fram till dem.

Vanligast är att länka till innehållets inledande rubrik. Denna kan utformas så här:

```
<h1 id="content" tabindex="-1">Innehållets huvudrubrik</h1>
```

Tabindex="-1" används för att få ett korrekt uppförande i alla webbläsare.

Länkar till innehållet skapas som vanliga länkar men med tecknet # följt av målets id-värde. Exempelvis:

```
<a href="#content" accesskey="s">
  Hoppa till textinnehållet
</a>
```

Skärmbilden nedanför visar ett exempel på en genväg till sidans huvudinnehåll på riksdagens webbplats.

Till sidans huvudinnehåll

SVERIGE
RIKSD

Start

L

Kommentar

Innehållet på mobilt är så pass koncentrerat att det sällan finns en anledning att behöva hoppa över innehåll, punkten bedöms därför som ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.1 \(A\)](#)

[WCAG 2.1 - 2.4.1 \(A\)](#)

[Vägledning för webbutveckling - R75 \(1\)](#)

[EN 301 549 - 9.2.4.1](#)

NL70: Snabbkommandon aktiveras inte bara med alfanumeriska tangenter

– Inte aktuellt

Bakgrund

Många användare styr gränssnittet helt eller delvis med tangentbordsnavigation och snabbkommandon. För en del användare med motoriska funktionsnedsättningar eller grava synnedsättningar är tangentbordet det primära sättet att styra gränssnittet. Även röststyrning innebär i många situationer att röstkommandon omvandlas till tangentbordskommandon, exempelvis kan en person som styr gränssnittet med rösten säga "tabb" för att hoppa till nästa länk, något som innebär att hjälpmedlet skickar tangentbordskommandon "tabb" till gränssnittet.

Det är viktigt att genvägar och snabbkommandon inte bygger på endast siffror, bokstäver eller symboler eftersom det kan leda till att användare som navigerar med rösten eller tangentbordet av misstag råkar aktivera funktioner. Exempel på det kan vara att information raderas eller att användaren ringer upp kontakter av misstag.

Om den här typen av snabbkommandon används så måste något av följande vara uppfyllt:

- Det finns en mekanism för att stänga av snabbkommandon.
- Snabbkommandon går att ändra, så att de styrs av en annan tangent.
- Snabbkommandot är bara möjligt att använda när objektet som styr kommandot är i fokus.

Html-attributet accesskey omfattas inte av denna riktlinje eftersom den typen av snabbkommandon aktiveras tillsammans med Ctrl-, Alt- eller Commandtangenter.

Kommentar

Var har ej sett inslag av snabbkommandon och punkten bedöms därför som ej aktuell

Kopplade riktlinjer

WCAG 2.1 - 2.1.4 (A)

EN 301 549 - 9.2.1.4

NL90: Påbörjad klickhändelse ska kunna avbrytas eller ångras

✓ Inga fel funna

Bakgrund

Ett klick består av två delar, dels en ner-händelse och dels en upp-händelse. Ner-händelsen sker när användaren trycker ner sin musknapp eller sitt finger på skärmen. Upp-händelsen är när knappen släpps eller fingret lyfts från skärmen.

När användare navigerar i ett gränssnitt genom att klicka sig fram med musmarkör eller med en pekskärm, inträffar ibland oavsiktliga händelser, användaren trycker helt enkelt på fel ställe. Genom att undvika funktionalitet som sker när användaren klickar ner minskar de problemen. Låt istället funktionaliteten aktiveras när användaren släpper musknappen.

I vissa fall är det inte möjligt, exempelvis för en piano-app, där användaren spelar på pianot genom att klicka ner på tangenterna.

Om en händelse måste aktiveras genom en ner-händelse så ska det om möjligt finnas en chans för användaren att ångra händelsen. Ett exempel på detta är många skärmtangentbord för mobiler. Genom att låta fingret vila på en bokstav dyker ett antal mindre vanliga bokstäver upp som baserar sig på den bokstav användaren vilat fingret över (exempelvis kan du skriva á â om du vilar med fingret över A-tangenten). Om användaren släpper skärmen utan att välja något skrivs inte heller något tecken.

Sök resa

Nästa avgång

Från station/hållplats/adress

Använd min plats

Till station/hållplats/adress

1234567890

Qß§ŠšŞIOPÅ

ASDFGHJKLÖÄ

↑ZXCVBNM↵

!#😊😊 Svenska Gå

Kommentar

Aktivering av en funktion sker när fingret lyfts upp som det bör, punkten blir godkänd.

Kopplade riktlinjer

[WCAG 2.1 - 2.5.2 \(A\)](#)

[EN 301 549 - 9.2.5.2](#)

NL100: Innehåll och funktioner ska kunna användas utan svåra gester

✓ Inga fel funna

Bakgrund

Användare med nedsatt rörlighet i händer och fingrar har svårt att kontrollera den precision som ofta krävs för att styra ett gränssnitt. Det finns också inmatningstekniker där det inte går att göra komplexa gester, som ögonstyrning. Exempelvis är det mycket svårt eller omöjligt att göra gester som kräver mer än en beröringspunkt med ögonstyrning, och att åstadkomma en rörelsebana med brytarstyrning.

Det är därför viktigt att skapa gränssnitt som bygger på så enkla gester som möjligt, utöver de gester som krävs för att styra operativsystem och webbläsare. Det ska finnas alternativa sätt som gör det möjligt för dessa användare att enkelt styra gränssnittet utan komplexa gester. Användare med huvudmus eller ögonstyrning är inte hjälpta av kortkommandon via ett skärmtangentbord. Gester ska istället kompletteras med knappar som användaren kan trycka på.

Exempel på en typ av gränssnitt som bygger på gester är serverbaserade kartor där användaren genom att knipa ihop eller isär med två fingrar zoomar in respektive ut i en karta. En annan gest är att med två fingrar i sid- eller höjdlid flytta det geografiska området som visas i kartan. Alternativet för användare med nedsatt motorik är att zoomningsfunktionen kan styras med knappar: plus för att zooma in och minus för att zooma ut. Knappar för höger, vänster, upp och ner låter användaren titta och panorera i kartan för att ändra det geografiska området.

Kommentar

Vi finner inga svåra gester i appen och bedömer punkten som godkänd.

Kopplade riktlinjer

WCAG 2.1 - 2.5.1 (A)

EN 301 549 - 9.2.5.1

NL110: Synliga beskrivningar av klickbara objekt beskrivs på liknande sätt genom uppmärkning i koden

✓ Inga fel funna

Bakgrund

Det finns många tekniker för att beskriva interaktiva objekt. För vanliga inmatningsfält används ofta en synlig ledtext i anslutning till fältet. Detta räcker oftast om ledtexten är inlagd på rätt sätt, men i vissa situationer används också visuellt dolda beskrivningar inlagda med exempelvis aria-label. Det kan finnas många skäl till att lägga in en visuellt dold beskrivning. Det kan exempelvis handla om att en användare som inte ser gränssnittet behöver en utökad beskrivning av fältet.

När dolda beskrivningar används är det dock viktigt att de inkluderar även det som visas visuellt. Ett objekt med en etikett som innehåller texten "Köp" eller "Läs mer", eller en bild som illustrerar objektets funktion, ska i den visuellt dolda etiketten eller beskrivningen inledas med samma text. En knapp med beskrivningen "Köp biljett" ska exempelvis också inledas med formuleringen "Köp biljett" i attributet aria-label eller aria-labelledby.

Användare med kognitiva eller grava motoriska funktionsnedsättningar kan använda rösten för att styra ett gränssnitt, istället för att använda tangentbord eller pekdon för att navigera. När visuellt synliga och dolda etiketter stämmer överens fungerar även röststyrning på ett bra och förutsägbart sätt.

Kommentar

Alla knappar och reglage som vi stöter på har en likvärdig beskrivning visuellt som med skärmläsaren, detta är avgörande för att användare ska ha en möjlighet att navigera runt i gränssnittet med hjälp av röststyrning.

Kopplade riktlinjer

[WCAG 2.1 - 2.5.3 \(A\)](#)

[EN 301 549 - 9.2.5.3](#)

NL120: Funktionalitet som aktiveras eller styrs genom rörelse kan inaktiveras och styras på andra sätt

– Inte aktuellt

Bakgrund

Mobila enheter kan förutom tangentbord och tal, även använda accelerometer, gyroskop och kamera som metod för inmatning från användaren, eller för att utföra en funktion i enheten.

För användare med motoriska funktionsnedsättningar är det problematiskt när funktioner i ett gränssnitt är baserade på rörelse. Det förutsätter att användaren kan röra fritt på sin enhet vilket exempelvis inte är möjligt för användare som har sin enhet monterad på sin rullstol. Om användaren inte erbjuds ett alternativ kan det vara omöjligt för vissa användare att styra ett gränssnitt. Därför ska gränssnitt, när det är möjligt, kunna styras med knappar, länkar och andra objekt som alternativ till rörelse.

Det finns undantag för funktioner som är beroende av rörelse för att fungera, exempelvis stegräknare och vattenpass.

Kommentar

Vi har ej stött på funktionalitet som styrs genom rörelse.

Kopplade riktlinjer

[WCAG 2.1 - 2.5.4 \(A\)](#)

[EN 301 549 - 9.2.5.4](#)

NL140: Markering av klickbara objekt och intilliggande färger ska ha en kontrast på minst 3,0:1

× Underkänd

Bakgrund

Det är inte bara text som behöver vara tydlig och ha tillräckliga kontraster i ett gränssnitt. Interaktiva objekt som knappar, länkar och formulärfält ska vara tydligt urskiljbara, även i dåliga ljusförhållanden. För att även användare med lättare synnedsättningar ska kunna urskilja gränssnittskomponenter, är det viktigt att ramar kring sådana objekt har goda kontraster mot intilliggande färger. Här ska kontrastvärdet vara minst 3,0:1.

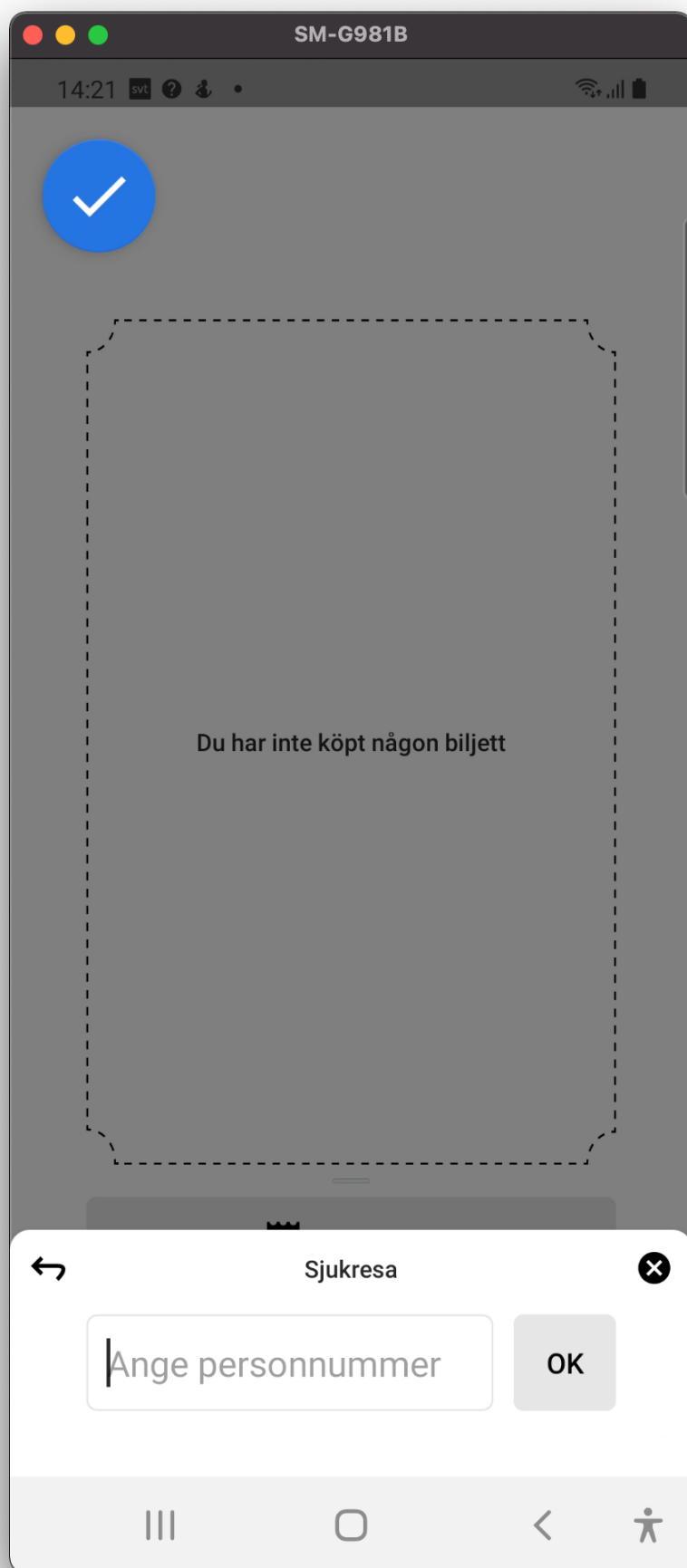
Kontrastkravet mellan ram- och bakgrundsfärg gäller i flera situationer: när

användare markerar ett klickbart objekt eller formulärsobjekt, när ett objekt är i fokus när användaren styr med tangentbordet eller när användaren har valt ett specifikt objekt.

Kontrastkravet gäller inte komponenter vars utformning inte har anpassats eftersom utseendet på dessa bestäms av användarens webbläsare. Det gäller inte heller inaktiva komponenter.

Kommentar

Ramen för inmatningsfältet mäter 1,2:1 vilket inte är godkänt mot kravet på 3,0:1.



Rekommendationer

- Öka kontrasten på ramen runt för inmatningsfältet till minst 3,0:1. Kontrollera med verktyget Color Contrast Analyzer att rätt kontrast är uppnådd.

Kopplade riktlinjer

WCAG 2.1 - 1.4.11 (AA)

EN 301 549 - 9.1.4.11

NL170: Länkgrupper och informationsområden är grupperade

– Inte aktuellt

Bakgrund

För att underlätta för användaren att identifiera grupper med objekt är det viktigt att det finns en tydlig gruppering i koden, exempelvis genom att menyer grupperas med hjälp av punktlistor. Då presenteras menylänkarna som en sammanhållen grupp även i hjälpmedel för gravt synskadade användare. I menyer där undermenyer är utfällda bör nästlade listor användas (listor i listor). Då blir även den hierarkiska strukturen möjlig att uppfatta för gravt synskadade användare.

I html5 finns elementet nav som ska användas för att ange vad som är navigation på sidan. Elementet ska omsluta både huvudmeny och undermeny, eller så ska två olika nav-element användas för att markera huvudmeny respektive undermeny. Elementet kan även användas för en större meny i sidfoten. Så här används det:

```
<nav> ... menylänkar ... </nav>
```

Landmärken

Wai-aria definierar olika landmärken (eller "landmarks" på engelska), som också används för att märka upp olika typer av innehåll på en sida. Dessa tillför ytterligare struktur för användare som inte ser gränssnittet.

Om du bygger med html5 ska du använda de element som finns i html5 i första hand. Du kan även lägga på olika roller, landmärken, för olika områden men det kan upplevas som störande om samma information finns i html-elementet.

Det vi menar här att <nav> aldrig ska kompletteras med role="navigation", <main> ska inte kompletteras med role="main" och <header> respektive <footer> ska inte kompletteras med role="banner" eller role="contentinfo".

Om båda används kan skärmläsare läsa upp informationen två gånger vilket blir störande för användaren.

Så här lägger man in landmärken:

```
<div role="search"> ... sökfunktion ... </div>
```

Använd följande landmärken där det är relevant:

- **role="search"**, som markerar sökfunktionen
- **role="main"** eller **<main>**, som markerar huvudinnehållet på sidan
- **role="complementary"**, som används för innehåll som kompletterar huvudinnehållet på sidan. Dit hör alla typer av "relaterad information", det vill säga den typen av innehåll som typiskt brukar ligga i högerspalten på sidan.
- **role="banner"** eller **<header>**, som används för sidhuvudet
- **role="contentinfo"** eller **<footer>**, som används för information om innehållet, exempelvis ett område med copyright-information och länkar till sekretessinformation.

Det finns fler landmärken:

- **role="application"**, som används för områden med applikationsliknande interaktion, exempelvis en texteditor. Undvik detta landmärke om du inte har mycket starka skäl till att använda det.
- **role="form"**, undvik detta eftersom det inte tillför något för användarna i dagsläget.

[Information om wai-aria \(W3C:s webbplats\)](#)

.

Kommentar

Vi kan ej se att det finns eller inte finns informationsområden i appen, punkten bedöms som ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R27 \(3\)](#)

[EN 301 549 - 9.1.3.1](#)

NL190: Länkars mål framgår tydligt i sitt sammanhang

! Inga fel funna men kan förbättras

Bakgrund

Tydligt formulerade länktexter hjälper alla användare att hitta rätt länk. En tydligt formulerad länk är också ofta en förutsättning för att gravt synskadade användare och en del användare med kognitiva funktionsnedsättningar ska hitta den information de söker. Länkar måste därför ge tydlig information om vilka sidor de leder till och vad det finns för typ av innehåll på de sidorna.

För en gravt synskadad användare är det omöjligt att få en visuell överblick av sidan. Det innebär att det inte med ett snabbt ögonkast går att identifiera de länkar som finns på sidan och vilka textstycken de är kopplade till. Gravt synskadade användare får istället ta sig runt på sidan och se vad som finns med stöd av sitt hjälpmedel, en så kallad skärmläsare.

Skärmläsaren presenterar bara den information användaren har fokus på. Är fokus på en länk läses bara länken upp, inte texten runt om. Användaren kan inte visuellt koppla länken till någon text i närheten. Det gör att länken i sig måste ge all relevant information om var den leder.

Många hjälpmedel har dessutom en funktion som lyfter länkarna ur sitt sammanhang och presenterar dem i en separat länklista. Detta snabbar upp navigationen och gör det möjligt att snabbt hitta specifika länkar som "Logga in" eller "Kundvagn". När en länk tas ur sitt sammanhang hjälper det inte användaren om det bredvid länken står var den leder, utan länkens mål måste framgå av själva länktexten. Länkar som heter "Läs mer" är därför inte tillräckliga.

Länkade bilder måste ge all relevant information om länken i alt-texten (kallas även det ibland för "beskrivning" eller "tooltip" i olika publiceringsverktyg).

För e-postlänkar är det viktigt att själva e-postadressen framgår i länktexten och även information om mottagaren, exempelvis mottagarens funktion och/eller namn. Att e-postadressen framgår är viktigt för att användare som lånat en dator eller inte vill e-posta just nu, ska kunna kopiera adressen för att använda vid ett senare tillfälle.

Title-text

Lägg inte någon title-text (kallas ibland för "beskrivning" eller "tooltip") på länkar. Länktexten ska ge användaren all relevant information om länken. Tidigare har en del webbplatser kompletterat länktexterna med en title-text som visas när

användaren håller muspekaren över länken. Den här lösningen går dock inte att ta till sig för exempelvis användare som navigerar med tangentbordet eller med en pekskärm.

Länkar är begripliga i sitt sammanhang

Wcag på nivå AA, kräver att länkarna är begripliga i sitt programmatiskt bestämbara sammanhang. Det innebär att länken ska gå att förstå tillsammans med annan text i samma html-element.

Det kan innebära att ett textstycke och en länk ligger i samma paragraf-element eller i samma list-element för att ta ett par exempel.

Det lättaste sättet att uppfylla kravet är också det som blir enklast för användarna; skriv länktexter som gör det enkelt att förstå länkens syfte oavsett sammanhang.

Kommentar

Vi upplever att länktexter är tydligt formulerade. Den förbättring vi önskar att se är att länkarna under inställningar som öppnas i ett nytt fönster, också blir uppmärksammade med en ikon och förklaras för en skärmläsare.

Rekommendationer

- Komplettera med en ikon för de länkar som öppnas i webbläsaren.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.4 \(A\)](#)

[WCAG 2.1 - 2.4.4 \(A\)](#)

[Vägledning för webbutveckling - R5 \(1\)](#)

[EN 301 549 - 9.2.4.4](#)

NL360: När samma meny eller länksamling visas på olika sidor ska sorteringen vara konsekvent om det inte skulle uppfattas som ologiskt av användarna

✓ Inga fel funna

Bakgrund

Ändra inte sorteringsordningen i listningar av länkar, verktyg eller menyer på olika sidor. Samma meny eller länksamling ska presenteras i samma ordning på alla ställen om det inte skulle upplevas som ologiskt för användaren.

Kommentar

Vi finner inget avvikande beteende och punkten blir godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 3.2.3 \(AA\)](#)

[WCAG 2.1 - 3.2.3 \(AA\)](#)

[Vägledning för webbutveckling - R28 \(3\)](#)

[EN 301 549 - 9.3.2.3](#)

Färger & presentation

CP10: Texters förgrunds- och bakgrundsfärger ger tillsammans tillräckliga kontraster

× Underkänd

Bakgrund

För att besökaren ska ta del av innehållet på ett enkelt sätt är det viktigt att kontrasterna (egentligen ljuskontrast) mellan text och bakgrund är bra. På en kontrastskala som går från 1:1 (ingen skillnad) till 21:1 (svart på vitt) ska kontrasten vara minst 4,5:1.

För större text, som rubriker, räcker en kontrast på minst 3:1. Men vår rekommendation är ändå att hålla minst 4,5:1 för att texten ska vara tydlig för så många användare som möjligt. Stor text innebär 18 punkter/24 pixlar vanlig stil eller 14 punkter/18.5 pixlar fet stil.

För att mäta kontraster används en komplex formel som finns på W3C:s webbplats:

[Formel för kontraster \(W3C:s webbplats, engelska\)](#)

.

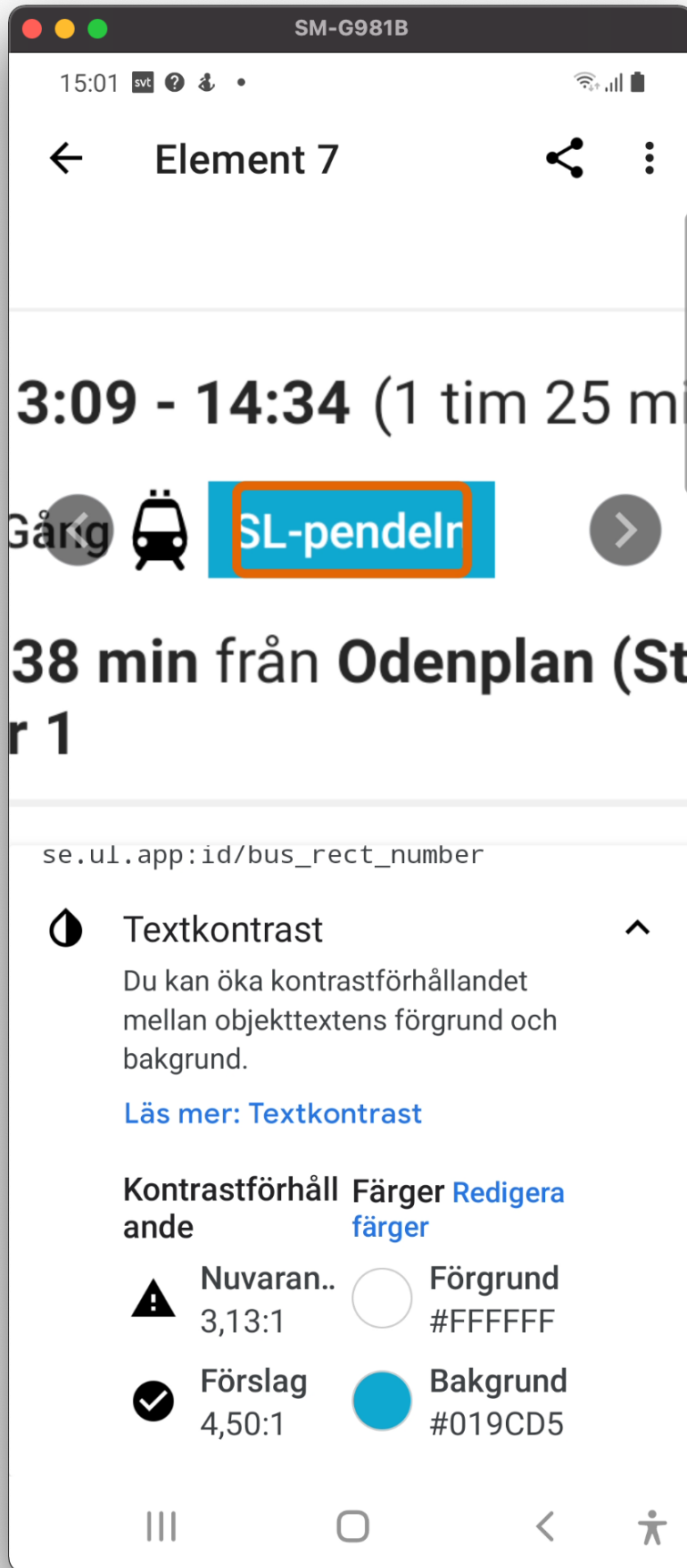
Om du inte själv vill sitta och räkna på detta rekommenderar WAI bland annat verktyget "Contrast Analyser" som kan laddas ned kostnadsfritt från Paciello Groups webbplats:

[Contrast Analyser \(the Paciello group webbplats, engelska\)](#)

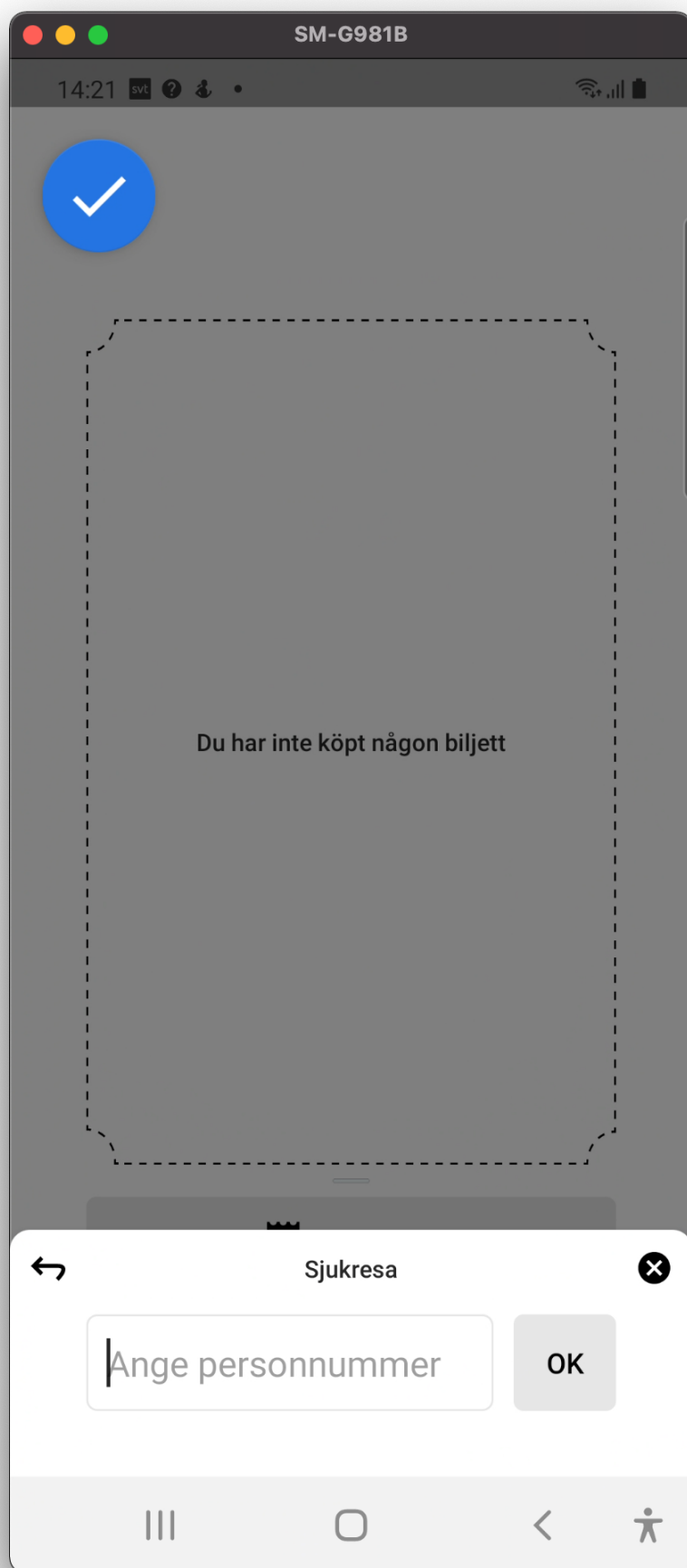
.

Kommentar

Vi har sett inslag där text inte når upp till kravet om 4,5:1 på kontrastskalan. Texten inuti taggen "SL-pendeln" är inte tillräcklig och uppnår 3,13:1 när vi mäter.



Placeholder texten är ett annat exempel med för låg kontrast. Här mäter den 1,2:1.



Rekommendationer

- Säkerställ att text uppnår minst 4,5:1 i kontrastvärde.

Kopplade riktlinjer

WCAG 2.0 - 1.4.3 (AA)

WCAG 2.0 - 1.4.6 (AAA)

WCAG 2.1 - 1.4.3 (AA)

WCAG 2.1 - 1.4.6 (AAA)

Vägledning för webbutveckling - R34 (1)

Vägledning för webbutveckling - R126 (1)

EN 301 549 - 9.1.4.3

CP20: Grafiska objekt som förmedlar information, och texter som förekommer i bilder, illustrationer och filmer har tillräckliga kontraster

✓ Inga fel funna

Bakgrund

Kontrasterna i illustrationer, diagram, bilder, ikoner och filmer ska vara bra. Användaren ska kunna se och tolka den grafiska informationen även i dåliga ljusförhållanden och med en lättare synnedsättning. Sträva därför efter så goda kontraster som möjligt. Gränsvärdet när det gäller kontraster i den här typen av material ska vara minst 3:1 enligt WCAG 2.1. Värdet avser färgen på det grafiska motivet mot bakgrunden. Ett undantag för kravet är när motivet är kompletterat med text.

Kommentar

Kontrasterna på ikoner uppfyller kraven.

Kopplade riktlinjer

WCAG 2.1 - 1.4.11 (AA)

EN 301 549 - 9.1.4.11

CP50: All information går att nå när användaren ställt in större avstånd mellan bokstäver, ord, rader eller stycken

– Inte aktuellt

Bakgrund

För användare med dyslexi och nedsatt syn är det viktigt att kunna anpassa hur texterna visas. Det finns olika plugins för webbläsare som gör det möjligt att styra storlek, typsnitt och avstånd mellan rader, ord, tecken och stycken. Ändringar av avstånd i text ska kunna göras utan att funktioner blir oåtkomliga eller att innehållet döljs eller klipps av. Resultatet måste inte vara perfekt, men ändringar ska vara möjliga att utföra i användarens webbläsare och ska inte blockeras. Användaren ska kunna ändra avstånd enligt de mått som anges nedan.

- Linjehöjd (line height) satt till minst 1.5 gånger fontens storlek
- Avståndet efter textstycken satt till minst 2 gånger fontens storlek
- Avståndet mellan bokstäver (letter spacing) satt till minst 0.12 gånger fontens storlek
- Avståndet mellan ord (word spacing) satt till minst 0.16 gånger fontens storlek

Kommentar

Denna funktion saknas i dagsläget i appar och punkten blir ej akutell.

Kopplade riktlinjer

WCAG 2.1 - 1.4.12 (AA)

EN 301 549 - 9.1.4.12

CP60: Webbplatsen presenteras utan störande skärmflimmer och utan rörliga och blinkande element som ej går att stänga av

✓ Inga fel funna

Bakgrund

Precis som en orolig bakgrund till text kan störa uppmärksamheten kan även rörelser i gränssnittet dra till sig besökarens uppmärksamhet och göra det svårare att koncentrera sig på texten. Därför bör rörelser, blinkningar och flimmer undvikas. Om man vill ha detta ska det antingen stanna automatiskt inom 3 sekunder eller ska det finnas ett tydligt sätt att stanna rörelsen.

Notera också att blinkningar som är snabbare än 3 gånger per sekund aldrig är

tillåtna.

Kommentar

Vi finner inget skärmflimmer när vi navigerar i appen och punkten blir godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 2.2.2 \(A\)](#)

[WCAG 2.0 - 2.3.1 \(A\)](#)

[WCAG 2.0 - 2.3.2 \(AAA\)](#)

[WCAG 2.1 - 2.2.2 \(A\)](#)

[WCAG 2.1 - 2.3.1 \(A\)](#)

[WCAG 2.1 - 2.3.2 \(AAA\)](#)

[Vägledning för webbutveckling - R133 \(1\)](#)

[EN 301 549 - 9.2.2.2](#)

[EN 301 549 - 9.2.3.1](#)

CP70: Sidorna har unika och relevanta sidtitlar

✓ Inga fel funna

Bakgrund

Sidans titel är det första som möter användaren. Den används också som text när användaren lägger in sidan bland sina favoriter och sökmotorerna använder sidtiteln när de indexerar sidor. Det är därför viktigt att webbplatsens alla sidor har unika sidtitlar som beskriver sidans innehåll och funktion på ett tydligt sätt.

Vi rekommenderar att du först i titeln anger sidans unika innehåll och att du därefter anger organisationens-/webbplatsensnamn.

Kommentar

Sidor och fönster har uppläsande titlar, vi bedömer punkten som godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.2 \(A\)](#)

[WCAG 2.1 - 2.4.2 \(A\)](#)

[Vägledning för webbutveckling - R135 \(1\)](#)

[EN 301 549 - 9.2.4.2](#)

CP110: Sidans huvudsakliga språk är angivet i koden

✓ Inga fel funna

Bakgrund

Huvudsakligt språk ska anges i koden på samtliga sidor. Detta gör det möjligt för uppläsande hjälpmedel att avgöra på vilket språk texten är skriven och därmed välja rätt röst för det aktuella språket. Huvudsakligt språk ska anges direkt i elementet html, överst i sidans kod.

I html5 anges huvudsakligt språk med lang-attributet:

```
<html lang="sv">
```

I xhtml anges huvudsakligt språk med attributet xml:lang="sv". Detta ska dock kompletteras med det äldre sättet att skriva för bakåtkompatibilitet (undantaget xhtml 1.1). Exempel:

```
<html xml:lang="sv" lang="sv">
```

Kommentar

När vi byter språk i telefonen följer appen med.

Kopplade riktlinjer

[WCAG 2.0 - 3.1.1 \(A\)](#)

[WCAG 2.1 - 3.1.1 \(A\)](#)

[Vägledning för webbutveckling - R17 \(3\)](#)

[Vägledning för webbutveckling - R141 \(1\)](#)

[EN 301 549 - 9.3.1.1](#)

CP120: Språk som avviker från sidans huvudsakliga språk är angivet i koden

× Underkänd

Bakgrund

När språkändringar görs i innehållet ska dessa märkas ut. Det kan exempelvis vara om det finns ett citat på sidan som är på ett annat språk eller om sidan ger information på ett språk men menyerna och webbplatsens struktur är på ett annat. Sådana ändringar ska anges i koden. Detta kan göras på de flesta element, exempelvis span, li, div, a och p. Enstaka termer och ord behöver inte märkas upp.

I html5 anges språk med lang-attributet:

```
<p lang="en">
```

Detta anger att stycket i fråga är skrivet på engelska. Attributet kan användas på alla html-element i html5.

I xhtml anges språk med attributet xml:lang="en". Detta ska dock kompletteras med det äldre sättet att skriva för bakåtkompatibilitet (undantaget xhtml 1.1). Till exempel så här:

```
<p xml:lang="en" lang="en">
```

Kommentar

Som standard läser TalkBack upp innehåll i det språk som användaren har ställt in TalkBack på. Om appen har innehåll som avviker från det förinställda språket, kommer TalkBack att läsa upp det inkorrekt. I det här fallet uppläses alla svenska destinationer på engelska vilket gör uttalet inkorrekt och ibland väldigt svårt att uppfatta

Rekommendationer

Se att rätt språk är inställt på respektive element genom attributet nedan:

```
LocaleSpan
```

<https://developer.android.com/reference/android/text/style/LocaleSpan>

[Problem att visa HTML-tag: code](#)

[Problem att visa HTML-tag: code](#)

Kopplade riktlinjer

[WCAG 2.0 - 3.1.2 \(AA\)](#)

[WCAG 2.1 - 3.1.2 \(AA\)](#)

[Vägledning för webbutveckling - R17 \(3\)](#)

[Vägledning för webbutveckling - R142 \(1\)](#)

[EN 301 549 - 9.3.1.2](#)

Struktur & formatering

ST20: Områdesrubriker beskriver områdets funktion tydligt och kortfattat

✓ Inga fel funna

Bakgrund

Precis som rubriker i innehållet på sidan är det viktigt att områdesrubrikerna hjälper användaren. En områdesrubrik ska kort och så specifik som möjligt återge vad området har för innehåll. Generiska rubriker som "Sidfot", "Relaterat" eller "Läs mer" fungerar inte bra. Försök i stället att använda mer konkreta rubriker som "E-tjänster", "Broschyrer och fördjupning" eller "Kontakt".

Kommentar

Rubrikerna är tydliga och enkla att förstå, punkten bedöms som godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.6 \(AA\)](#)

[WCAG 2.1 - 2.4.6 \(AA\)](#)

[Vägledning för webbutveckling - R61 \(1\)](#)

[EN 301 549 - 9.2.4.6](#)

ST40: Innehållsrubriker har täckning i texterna och lockar till läsning

✓ Inga fel funna

Bakgrund

Rubriken är det första som läsaren läser. Det kan också bli det enda som läsaren läser, om rubriken inte hänger ihop med texten på ett korrekt sätt eller leder till "fel" innehåll.

Därför är det viktigt att rubrikerna ger relevant information om den tillhörande texten. Rubrikerna bör innehålla det eller de viktigaste nyckelorden i texten. De ska också vara inbjudande, så att de lockar läsaren att läsa mer.

Både huvudrubriker och underrubriker bör vara korta, men de ska inte vara som

en överskrift med bara ett eller två ord. Var hellre tydliga med vilket innehåll som kommer efter rubriken. Ofta är det bra att låta rubriker bestå av en hel mening, gärna med ett aktivt verb.

Kommentar

Rubrikerna för villkoren är tydliga och informativa, punkten bedöms som godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.6 \(AA\)](#)

[WCAG 2.1 - 2.4.6 \(AA\)](#)

[Vägledning för webbutveckling - R61 \(1\)](#)

[EN 301 549 - 9.2.4.6](#)

ST60: Text som visuellt fungerar som rubriker är också kodade som rubriker

× Underkänd

Bakgrund

Rubrikerna är något av det första som användaren ser och läser på en ny sida. Betydelsen av en bra rubriksättning är därför oerhört stor för att nå ut med informationen. Men alla användare kan inte se rubrikerna, en del är beroende av ett hjälpmedel för att ta till sig informationen. För att hjälpmedlen ska förstå vad som är rubriker på en sida och återge dessa till användaren krävs det att de är korrekt kodade med h-elementen.

Det är flera saker som är viktiga men en grundförutsättning är att det som ser ut som, och därför visuellt fungerar som, rubriker också är kodade med korrekta rubrikelement.

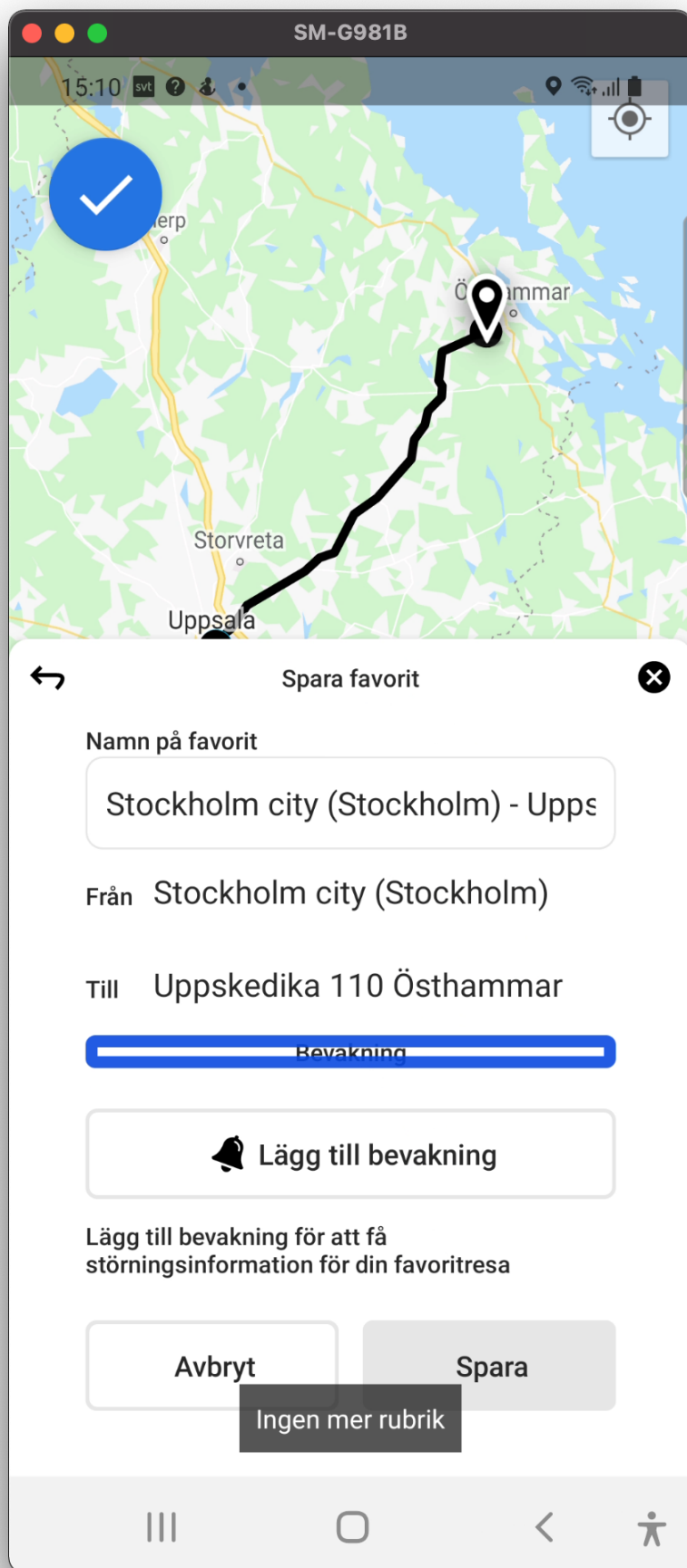
I html kodas rubriker med elementen h1, h2, h3, h4, h5 och h6. Siffran indikerar nivån. Det går också att koda rubriker med wai-aria. Då ger man texten role="heading" och anger nivå med attributet aria-level. Vi avråder dock från det eftersom stödet inte är hundra procentigt bland hjälpmedlen och det ökar risken för problem.

Det finns ett undantag till den här punkten. Om du har rubriker som visas så fort sidan laddas och som ligger före innehållets huvudrubrik i strukturen så ska de inte kodas som rubriker. Exempel på detta är en visuell rubrik ovanför vänstermenyn på en webbsida där navigationen kommer före innehållet i strukturen. Den ska inte kodas som en rubrik. Däremot om du har en utfällande

meny som bara visas när användaren klickat på en knapp eller länk, och den har visuella rubriker, då ska de vara kodade som riktiga rubriker.

Kommentar

Rubriker i appen är ej kodade som rubriker. Under villkoren finner vi dock fungerande rubriker men vi misstänker att det är en webb-vy som inte delar kod med appen.



Rekommendationer

Märk upp rubriker med följande och sätt värdet till "True":

```
setAccessibilityHeading
```

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.1.3.1](#)

ST80: Rubrikstrukturen inleds med en huvudrubrik.

– Inte aktuellt

Bakgrund

Rubrikelementen ska forma en logisk struktur. Det innebär att första rubriken ska vara kodad som en h1-rubrik.

Exempel:

```
<h1>Huvudrubrik</h1>
  <h2>Underrubrik</h2>
    <h3>Rubriknivå 3</h3>
    <h3>Rubriknivå 3</h3>
  <h2>Underrubrik</h2>
```

Kommentar

Då rubriknivåer inte finns på Android kan punkten ej bedömas.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R105 \(1\)](#)

[EN 301 549 - 9.1.3.1](#)

ST90: Rubrikstrukturen är logisk och återger innehållets hierarki

– Inte aktuellt

Bakgrund

Rubrikstrukturen ska vara logisk och spegla strukturen i sidans innehåll. Inled alla sidor med en korrekt huvudrubrik (elementet h1) överst i innehållet. Därefter ska alla rubriker komma i en hierarkiskt korrekt ordning som återspeglar innehållets struktur. Tänk på att siffran inte ska återge hur viktig rubriken är utan hur den förhåller sig till övriga rubriker, är det en underrubrik till föregående rubrik, eller ska den ligga på samma nivå?

Föregående punkt handlar om att strukturen är formellt korrekt. Den här punkten handlar om att det är en struktur som faktiskt matchar innehållet på sidan.

Kommentar

Då rubriknivåer inte finns på Android kan punkten ej bedömas.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R105 \(1\)](#)

[EN 301 549 - 9.1.3.1](#)

ST120: Listor är korrekt kodade och används på rätt sätt

! Inga fel funna men kan förbättras

Bakgrund

En del information passar att lägga in i listor. Det gäller exempelvis uppräkningslistor, definitionslistor och beskrivningar över olika steg i en process.

För att lägga in dessa listor ska de element som finns i html för ändamålet användas. Detta är en förutsättning för att ett hjälpmedel ska tolka och presentera dem som korrekta listor för användaren.

För att inleda en punktlista används elementet ul, för att inleda en numrerad lista används elementet ol. Varje listelement skapas sedan med elementet li.

För definitionslistor används elementet dl för att starta listan, dt för att ange den term som ska definieras och dd för att ge dess beskrivning.

Listor kan även användas för att strukturera menyn. Genom att lägga in menyn i en nästlad lista kan du förmedla menyens hierarki på ett rudimentärt sätt även för

användare som inte kan se den.

Exempel:

```
<ul>
  <li><a href="bs.html">Barn & skola</a></li>
  <li><a href="bb.html">Bo & bygga</a>
    <ul>
      <li><a href="blov.html">Bygglov</a></li>
      <li><a href="proj.html">Aktuella
byggprojekt</a></li>
    </ul>
  </li>
  <li><a href="tur.html">Turism</a></li>
</ul>
```

Kommentar

Vi har upptäckt att skärmläsaren får fokus på en lista utan innehåll på startsidan. Om listan inte behöver vara där kan den tas bort, alternativt inte kunna få fokus.

Rekommendationer

- Ta bort listan eller sätt följande till "False" så att fokus inte kan fås.

android:focusable

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R104 \(4\)](#)

[EN 301 549 - 9.1.3.1](#)

ST180: Stycken är skapade med elementet p och används på rätt sätt

✓ Inga fel funna

Bakgrund

För att lägga in textstycken på en sida ska elementet p användas. Detta gör dokumenten lättare att skumläsa med ett uppläsande hjälpmedel. Användaren kan då i många av dessa hoppa från stycke till stycke. Extra viktigt är detta givetvis i lagtexter där de olika styckena måste gå att särskilja från varandra. Använd inte elementet br eller hr för att skilja textstycken.

Kommentar

Text läses upp korrekt i appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.1.3.1](#)

ST200: Citat är markerade med q eller blockquote

– Inte aktuellt

Bakgrund

Märk upp längre citat med elementet blockquote.

Visuellt skapar elementet ett indrag i texten, något som en del publiceringssystem utnyttjar för att erbjuda en enkel funktion för indrag i texteditorn. Detta ska dock undvikas. Blockquote får bara användas för citat.

Använd elementet q för att märka upp och formatera kortare citat.

I både blockquote och q ska attributet cite användas för att ange en källa i form av en url om det finns en sådan källa.

Exempel:

```
<blockquote  
  cite="http://www.svt.se/vader/prognos_071224.html">
```

```
    <p>Det finns en 50-50 chans för snö i södra Sverige till  
    jul. Det bygger dock på att lågtrycket som nu ligger över  
    brittiska öarna styr norröver.</p>
```

</blockquote>

Kommentar

Vi hittar inga citat i appen och punkten blir ej aktuell.

Kopplade riktlinjer

WCAG 2.0 - 1.3.1 (A)

WCAG 2.1 - 1.3.1 (A)

Vägledning för webbutveckling - R1 (1)

EN 301 549 - 9.1.3.1

Bilder

IM10: Text presenteras som text, inte som bilder av text

✓ Inga fel funna

Bakgrund

Bilder är på många sätt ett bättre sätt att kommunicera information på än vanlig text. Med bilder kan man åskådliggöra komplexa samband och svåra begrepp. Bilder underlättar även i det att de bryter av texten och får användaren att uppleva texten som mindre intensiv och jobbig.

Bilder av text

Bilder ska alltså användas där det är lämpligt, men de ska inte användas för att representera text. Ska text läggas ut i innehållet, menyerna eller på andra ställen på sidorna, ska texten läggas ut som ren text. Bilder av text skapar många gånger kraftiga tillgänglighetsproblem.

När bilder förstoras blir varje bildpunkt större, men text är inlagd som vektorer, något som gör att varje tecken blir uppbyggt av fler bildpunkter när texten förstoras. Skillnaden blir då att bilden av text blir kantig och ofta upplevs som suddigare medan vanlig text förblir skarp. Detta blir extra tydligt i förstorande hjälpmedel. Nedanför har vi lagt in två bilder, först en bild som föreställer en bild av text vilken alltså är skapad med pixelgrafik och nedanför denna en bild på riktig text som alltså är skapad med vektorgrafik. Det är samma textstorlek och vi har sedan zoomat in 300 %.

Text som bild

Text som text

Ett annat problem med bilder av text är att texten i en bild inte kan anpassas av hjälpmedel för dyslektiker, den kan inte heller kopieras, något som en del hjälpmedel för denna grupp kräver. Även om man sätter en alt-text på bilden av text är det inte säkert att ett hjälpmedel för dyslektiker kan hantera detta.

Undantag

I vissa situationer kan det vara berättigat att använda text i en bild, det kan exempelvis gälla för organisationsscheman eller olika typer av flödesscheman där texten måste vara en del av bilden.

När text används i sådana bilder måste den göras minst dubbelt så stor som den vanliga texten på webbplatsen (detta eftersom texten i bilden inte kan förstöras på något bra sätt av webbläsare och hjälpmedel). Samtidigt måste kontrasterna vara bra, och bilden måste innehålla en alt-text.

Om inte denna beskrivning kan hålla all information som bilden visar måste det antingen finnas en utförlig beskrivning som läggs in via attributet longdesc eller en utförlig beskrivning i anslutning till bilden. Om det exempelvis rör sig om ett organisationsschema kan organisationen med dess olika avdelningar beskrivas i text under bilder. Använd rubriker och listor för att strukturera informationen på ett så tydligt sätt som möjligt.

Kommentar

Vi ser att text presenteras som text och punkten bedöms som godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 1.4.5 \(AA\)](#)

[WCAG 2.0 - 1.4.9 \(AAA\)](#)

[WCAG 2.1 - 1.4.5 \(AA\)](#)

[WCAG 2.1 - 1.4.9 \(AAA\)](#)

[Vägledning för webbutveckling - R128 \(1\)](#)

IM30: Likvärdiga textbeskrivningar finns för alla meningsbärande grafiska element på webbplatsen

Bakgrund

För att hjälpmedel ska återge innehållet i bilder till användaren måste bilderna förses med en alternativ beskrivning. Detta görs vanligen med hjälp av attributet alt (kan i olika publiceringssystem kallas för tooltip eller beskrivning). Den alternativa beskrivningen ska kortfattat redogöra för bildens motiv. Ju viktigare bilden är för sidan och förståelsen desto mer ingående behöver beskrivningen vara.

Alt-texten ska hållas under 150 tecken, men ibland räcker inte det för att återge all information i bilden. Då måste alt-texten kort ange vad bilden visar och hänvisa till en längre beskrivning eller till motsvarande information. Exempelvis:

"Diagram över befolkningsutvecklingen i Sverige. Se tabell efter bilden".

Meningsbärande grafiska objekt är i princip alla bilder som innehåller information, illustrationer, fotografier, ikoner, bildknappar, bildlänkar och så vidare. Icke meningsbärande grafiska objekt är exempelvis streck, tomrum, färgade bakgrundsplattor och liknande. Sådana bilder ska i första hand läggas in med css. Länkade bilder, kräver lite extra eftertanke. Om du har en länk som bara består av en bild så måste bildens alt-text i första hand återge länkens mål. Om du har en länk som både innehåller en bild och en text måste du tänka på att länkbeskrivningen kommer att bestå av länktexten och bildens alt-text. Beroende på situationer kan du då behöva ha en tom alt-text. Det är exempelvis bara irriterande att höra: *Skrivare Skriv ut*, om du har en skriv ut länk med ikon.

Även om du har formulär och där använder input type="image" så ska denna ha en alt-text som förklarar objektets funktion. Notera dock att det oftast är bättre att använda input varianter som är anpassade för syftet, exempelvis är det bättre att använda input type="submit" eller ett button-element för knappar, snarare än en bild.

Använder du elementen figure och figcaption så ska bilden fortfarande ha en alt-text, exempel:

```
<figure>
  <img ... alt="Sverige år 1790" />
```

```
<figcaption>
  Figur 1 – Bilden visar sveriges utbredning 1790...
</figcaption>
</figure>
```

Kommentar

I introduktionsguiden finns det illustrationer som ej får fokus och någon uppläsning från VoiceOver. Dessa räknas in som meningsbärande och skall ha en beskrivning.

Rekommendationer

Ange följande för att möjliggöra fokus på bilder och sätt värdet till "true" för meningsbärande bilder, eller false för dekorativa.

```
android:focusable
```

Ange sedan följande attribut för att lägga till en alt-text på meningsbärande bilder.

```
android:contentDescription
```

Kopplade riktlinjer

[WCAG 2.0 - 1.1.1 \(A\)](#)

[WCAG 2.1 - 1.1.1 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.1.1.1](#)

IM40: Alt-texten beskriver bilden eller bildens syfte

– Inte aktuellt

Bakgrund

Alla meningsbärande grafiska objekt ska innehålla en kort alternativ beskrivning (alt-text) som motsvarar bildens funktion i dokumentet. Alternativa beskrivningar ska:

- Beskriva bildens motiv
- Beskriva bildens funktion (ifall den är klickbar)

Bilder som huvudsakligen är av dekor-karaktär kan vara mycket kortfattat beskrivna. Bilder som förmedlar viktig information behöver beskrivas mer ingående.

Det är ofta bra om man beskriver vilken typ av grafiskt objekt som används. Till exempel: "Stefan Johansson från Funka Nu AB, foto" eller "Gubbe som spikar med en hammare, illustration".

Grafiska objekt som inte är meningsbärande (exempelvis layout-element) behöver inte beskrivas. Det är däremot viktigt att säkerställa att det finns tomma alt-attribut på sådana objekt.

Kommentar

Då bilderna saknar en alt-text kan vi ej bedöma deras beskrivning och punkten blir ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 1.1.1 \(A\)](#)

[WCAG 2.1 - 1.1.1 \(A\)](#)

[EN 301 549 - 9.1.1.1](#)

IM60: Klientbaserade bildkartor har textbeskrivningar för både länkar och bilder

– Inte aktuellt

Bakgrund

Bildkartor (eng. Imagemap) är bilder med klickbara områden i. Det finns två typer av bildkartor, klientbaserade och serverbaserade.

Klientbaserade bildkartor är i grund och botten bara en vanlig bild där ytor i bilden definierats som länkade områden. En serverbaserad bildkarta är en bild där du kan klicka var som helst i bilden och sedan skickas koordinaterna till servern som i sin tur bestämmer vad som ska skickas tillbaka till webbläsaren. Detta används exempelvis för många karttjänster där det går att zooma så som Google Maps och Eniro.

För användare med hjälpmedel utgör klientbaserade bildkartor ett mindre problem än serverbaserade bildkartor, det är en mycket enklare teknik och det är relativt enkelt att se till att alla områden får en lämplig beskrivning.

Tillgängliga, klientbaserade bildkartor

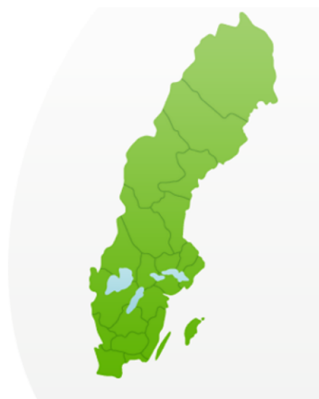
Den bild som används för bildkartan ska ha en kort beskrivning av vad bilden visar, precis som vilken meningsbärande bild som helst.

De olika klickbara områdena (area-elementen) ska beskrivas i attributet title, men kan som komplement även beskrivas i alt-attributet. Det går också som komplement att i anslutning till bilden lägga vanliga textlänkar till samma sidor

som länkarna i bilden leder till. Beroende på hur små de klickbara områdena är kan det rent av bli nödvändigt med en lista med länkar parallellt till bildkartan. Så här kan det se ut:

Ja tack! Visa mig även information från:

- | | | |
|---|------------------|-------------------|
| › Blekinge | › Kalmar län | › Värmland |
| › Dalarna | › Kronoberg | › Västerbotten |
| › Gotland | › Norrbotten | › Västernorrland |
| › Gävleborg | › Skåne | › Västmanland |
| › Halland | › Stockholms län | › Västra Götaland |
| › Jämtland Härjedalen | › Sörmland | › Örebro län |
| › Jönköpings län | › Uppsala län | › Östergötland |
| › Jag vill inte se specifik information från något landsting/region | | |



I exemplet kan användaren klicka på länet i listan eller i kartbilden till höger. Så här kan koden för en enkel klientbaserad karta se ut. Här har vi valt både att använda alt-text och title-text i de klickbara områden eftersom stödet för de båda attributen varierat i elementet area:

```

<map id="Karta" name="Karta">

<area shape="poly"
coords="102,0,73,12,44,74,27,88,29,115,45,136,61,152,93,157,1
32,158,137,60,132,10,120,0" href="www.funka.com" alt="Se
projekt i norra Sverige" title="Se projekt i norra Sverige">

<area shape="poly"
coords="25,117,56,152,91,161,143,164,144,201,79,193,33,184,18
,165" href="www.funka.com" alt="Se projekt i mellersta
Sverige" title="Se projekt i mellersta Sverige">

<area shape="poly"
coords="15,165,27,180,80,192,140,204,138,247,18,249,4,171"
href="www.funka.com" alt="Se projekt i södra Sverige"
```

```
title="Se projekt i södra Sverige">
</map>
```

Kommentar

Finns inga klientbaserade kartor och punkten bedöms som ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 1.1.1 \(A\)](#)

[WCAG 2.0 - 2.4.4 \(A\)](#)

[WCAG 2.0 - 2.4.9 \(AAA\)](#)

[WCAG 2.1 - 1.1.1 \(A\)](#)

[WCAG 2.1 - 2.4.4 \(A\)](#)

[WCAG 2.1 - 2.4.9 \(AAA\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.1.1.1](#)

[EN 301 549 - 9.2.4.4](#)

IM70: Serverbaserade bildkartor används enbart när det ökar användarnyttan och då erbjuds alternativa vägar att få fram motsvarande information på

× Underkänd

Bakgrund

Serverbaserade bildkartor kan ge ett stort mervärde till användarna. Det blir exempelvis mycket enklare att zooma ut och zooma in kartan där en adress är markerad och på så sätt orientera sig. Därför rekommenderar vi att du använder serverbaserade bildkartor när det finns ett mervärde för användaren. Det du måste tänka på är följande:

- Det ska gå att hantera kartan med tangentbordet
- Det ska finnas ett alternativt sätt att ta del av informationen som kartan förmedlar. Om det är en karta som visar var ett kontor ligger, ska adressen till kontoret också finnas i textform. Är det en karta med många olika intressepunkter måste det finnas en möjlighet att söka fram en intressepunkt och att lista intressepunkter i en textbaserad lista.
- Tillgängligheten i kartan behöver ses över. Det är omöjligt att inte till viss grad vara beroende av användarens förmåga att uppfatta färg i en karta, men

med en genomtänkt användning av ikoner, färger och symboler kan kartan göras rimligt tydlig för så många som möjligt.

- Ha en uttänkt strategi för hur kartan ska fungera i mobila enheter.

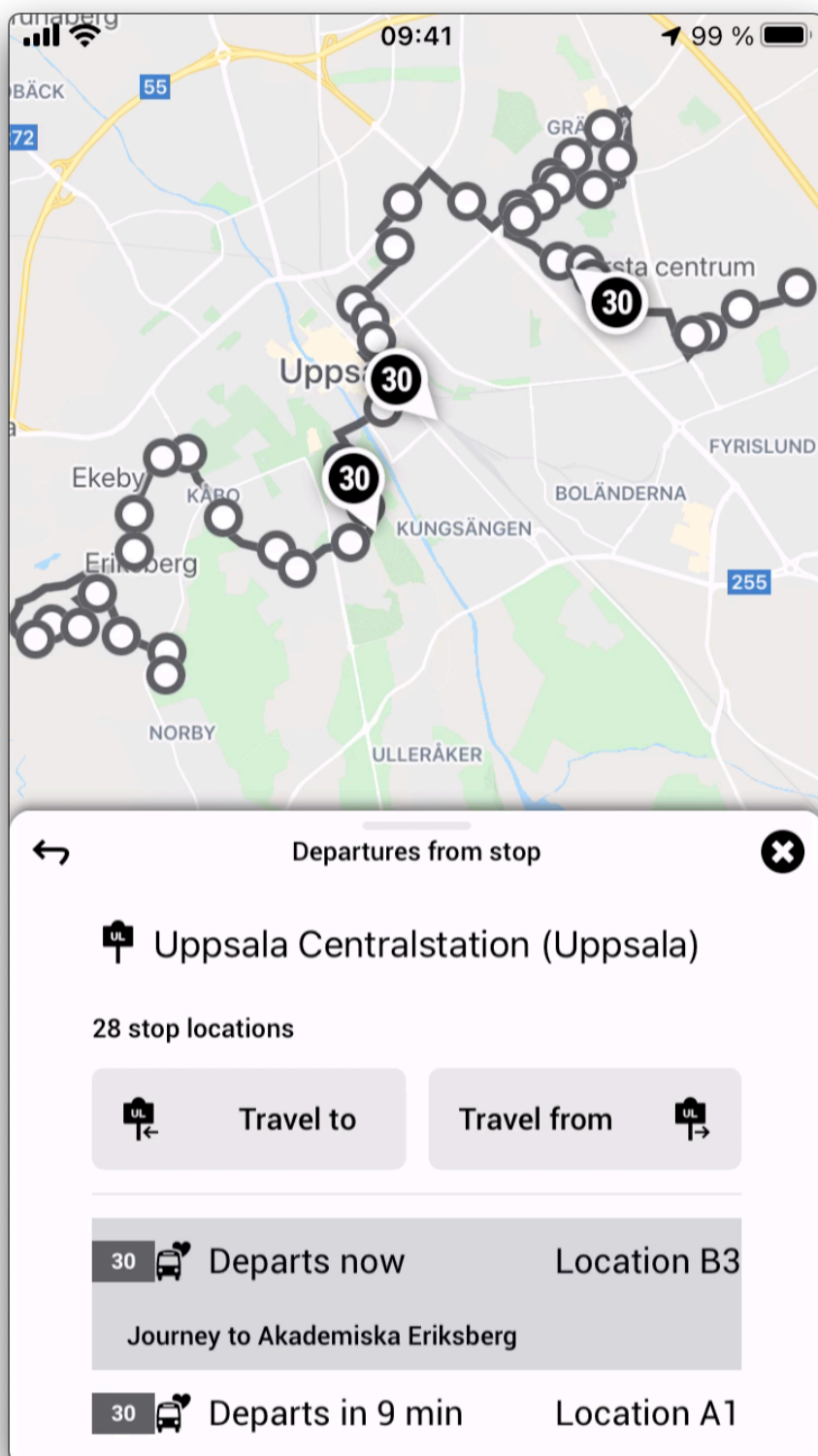
Användare som skrollar på pekskärm ska inte hamna i en oändlig skroll eftersom kartan tar hela fönstret, det måste gå att skrolla nedanför kartan på sidan (om inte kartan utgör hela sidan).

Kommentar

Kartor är undantagna lagen men innehållet i en karta skall presenteras på ett annat sätt som är tillgängligt. I UL-appen upplever vi att det är två kartfunktioner som inte presenteras på ett alternativt sätt. Dels är det live-funktionen att kunna se var ens buss är, samt information om de olika hållplatserna för en vald busslinje som visas i bilden nedan.

Live-funktionen är mycket svår att ersätta, och vad vi har sett finns det inget smidigt sätt som kan ersätta den informationen rakt av. Det finns några idéer på vad man kan göra och ett förslag är att presentera i textform, förutom när bussen ankommer i minuter, även vilken hållplats som bussen precis har lämnat eller ankommer till.

Den andra kartfunktionen behandlar information om alla hållplatser för en busslinje. Det skrivs ut hur många hållplatser som finns inom en busslinje, men information om vilka dessa är presenteras endast på kartan.



Skärmdumpen är lånad från iOS-appen, men vi ser att innehållet inte skiljer sig åt.

Rekommendationer

- Se över möjligheten att lägga till information för en skärmläseanvändare om var en buss befinner sig i detta nu.
- Skriv vilka hållplatser en busslinje har mellan start- och ändhållplats.

Kopplade riktlinjer

WCAG 2.0 - 1.1.1 (A)

WCAG 2.1 - 1.1.1 (A)

Vägledning för webbutveckling - R1 (1)

EN 301 549 - 9.1.1.1

Formulär

FS10: Formulärobject är kodade med korrekt formulärskod

× Underkänd

Bakgrund

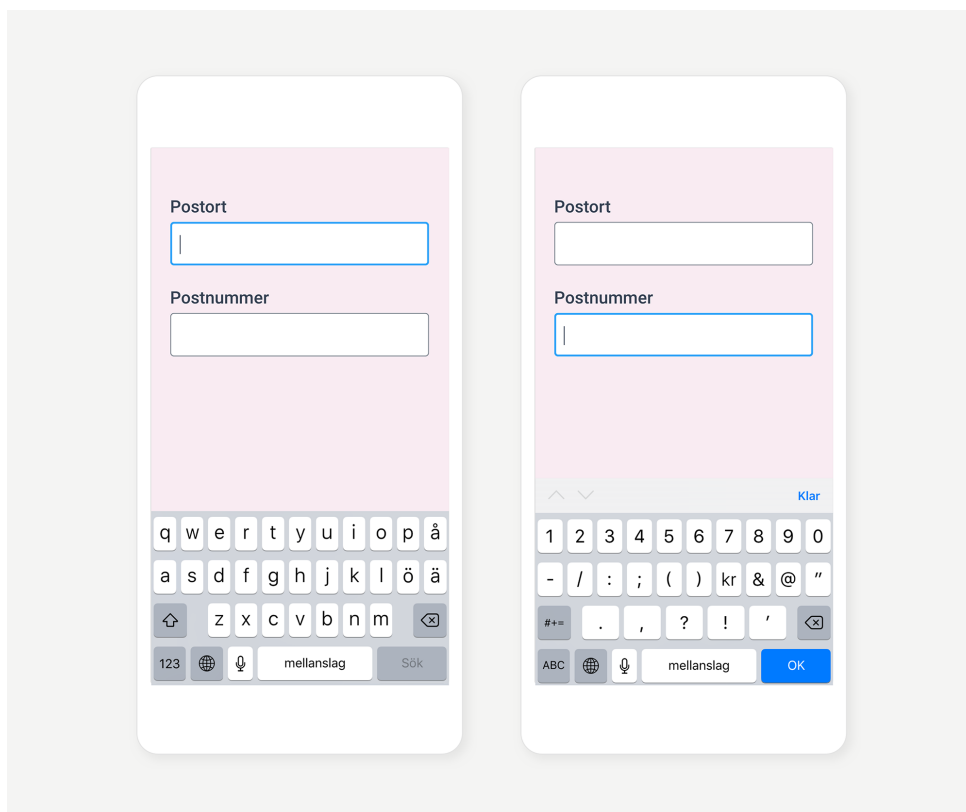
Det är viktigt att alla delar av ett formulär är inlagda som riktiga formulärobject då de annars inte presenteras tillsammans med övriga delar av formuläret i ett hjälpmedel. Ibland förekommer exempelvis knappar som är skapade som länkar (A-element) eller länkade bilder i stället för som korrekta formulärknappar. Då listas inte dessa bland övriga formulärobject och det blir onödigt svårt att skicka formuläret för en del användare.

Det är också viktigt att rätt typ av element används. I stället för att exempelvis använda kryssrutor men lägga in skript som hindrar att mer än en kryssruta är ikryssad åt gången så ska radioknappar användas.

I html5 finns flera nya typer av formulärobject. Dessa har många fördelar för användarna, inte minst i mobilen där tangentbordet kan anpassa sig efter den typ av information som ska matas in. Nedan ett exempel på hur olika typer av inmatningsfält ger upphov till olika tangentbord i mobilen.

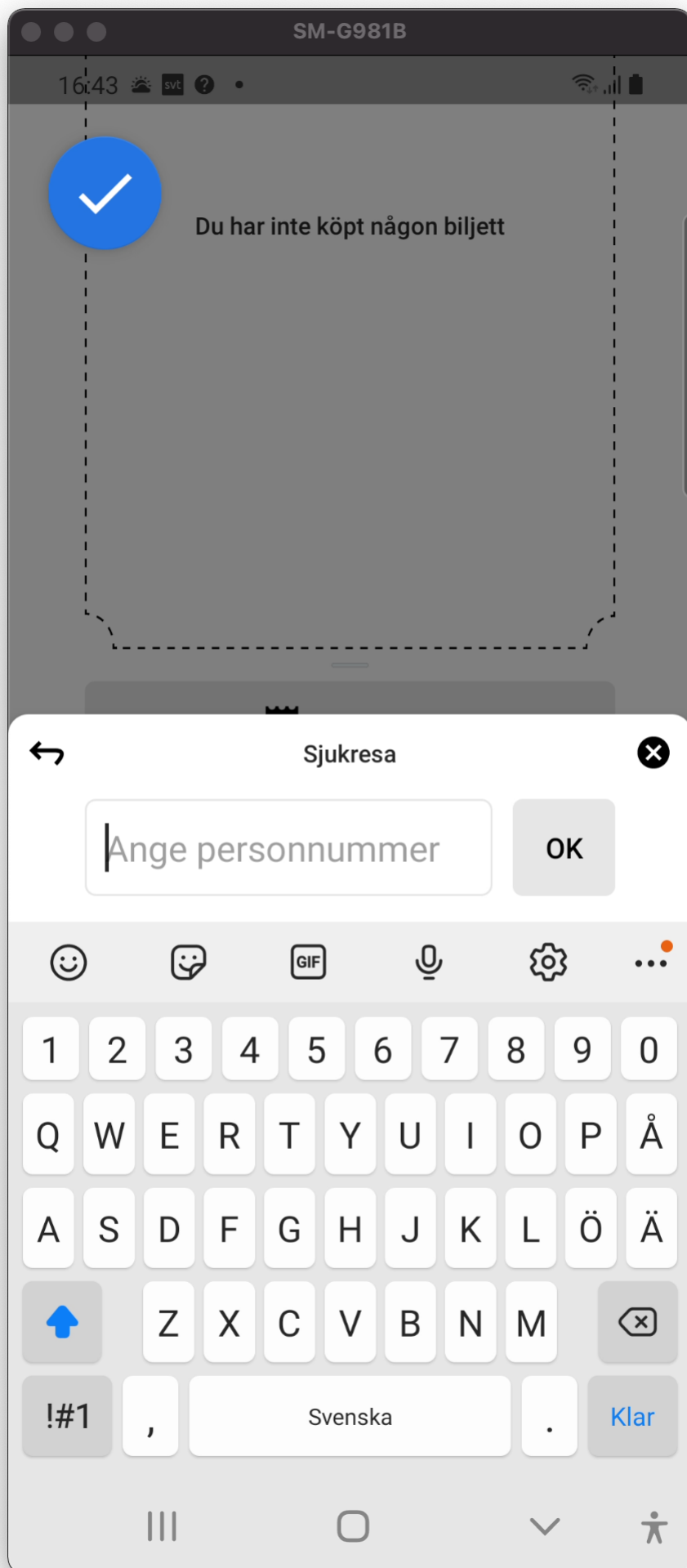
```
<label for="postnummer">Postnummer</label>
<input type="number" name="postnummer" id="postnummer">
```

```
<label for="stad">Stad</label>
<input type="text" name="stad" id="stad">
```



Kommentar

Det visuella tangentbordet skall matcha den data som efterfrågas. I exemplet nedan efterfrågas personnummer, då skall tangentbordet visa sifvertangenterna.



Rekommendationer

Säkerställ att rätt inmatningsfält används för den data som efterfrågas. Märk upp inmatningsfältet med följande:

```
android:inputType="phone"
```

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.0 - 4.1.2 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 4.1.2 \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

[EN 301 549 - 9.4.1.2](#)

FS50: Det finns ledtexter och instruktioner som beskriver för användaren hur och vad denne ska fylla i formuläret

× Underkänd

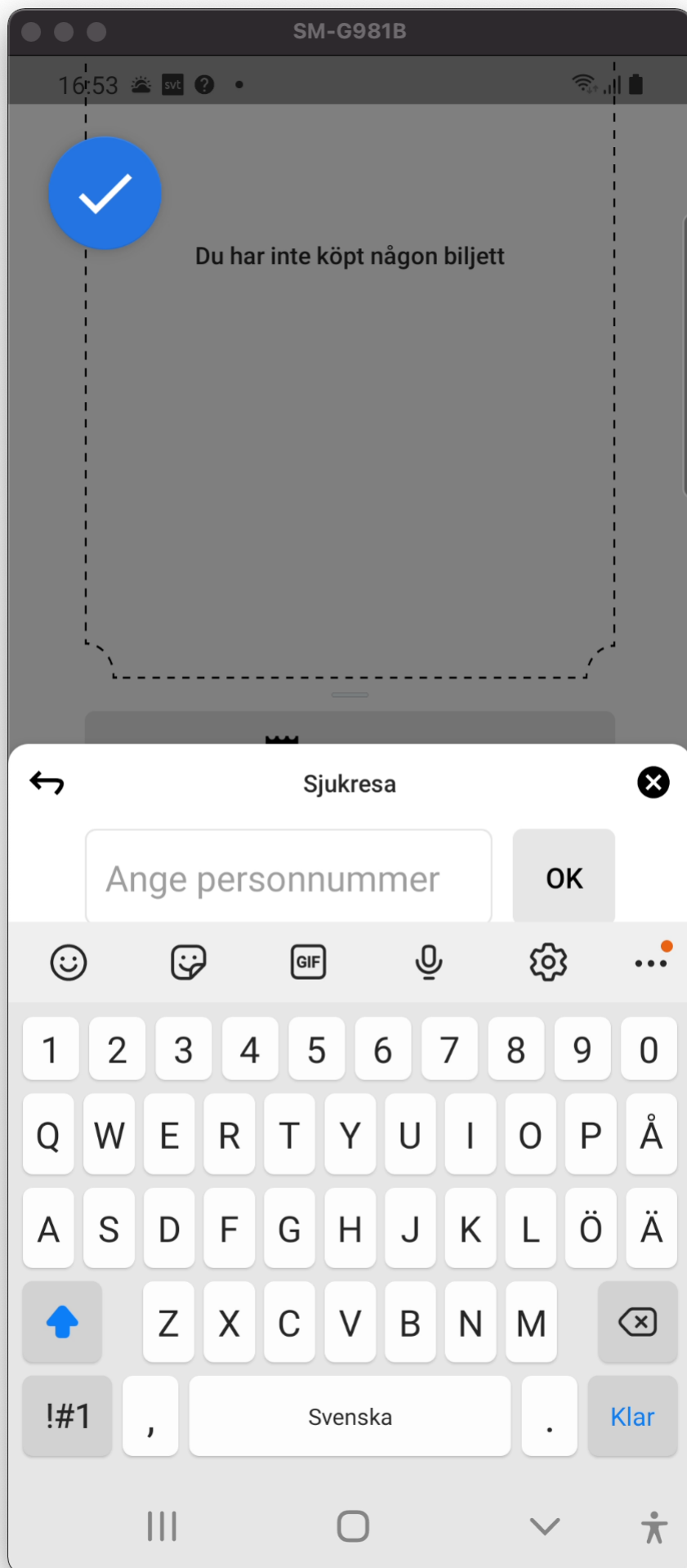
Bakgrund

När användaren ska välja eller mata in uppgifter måste det finnas ledtexter och instruktioner som tydligt anger vad som ska väljas eller matas in. Instruktionerna ska också innehålla information om vilket format uppgifter ska anges i och annan viktig information som användaren behöver för att kunna mata in information på rätt sätt.

Kommentar

Ledtexter förekommer i form av placeholder-text. Detta är i sig inte fel men placeholder-text tenderar att ha brister som låg kontrast samt att det försvinner vid fokus av inmatningsfältet vilket kan göra det svårt att komma ihåg ledtexten.

Ett annat nedslag är att det inte förekommer instruktioner på hur användaren skall skriva in den information som önskas. Som i exemplet nedan där ett personnummer efterfrågas. Exempel på instruktioner är att skriva ut vilket format som efterfrågas, tio eller tolv siffror. Ett annat exempel är för inmatning av värdekoder. Här kan det vara hjälpsamt att veta hur många tecken som en sådan värdekod brukar vara.



Rekommendationer

- Skriv ut ledtexten ovanför inmatningsfältet så att den alltid är synlig.
- Skriv ut vilket format som efterfrågas eller hur många tecken som förväntas.
- Säkra att denna information också kan läsas upp för skärmläseanvändare.

Kopplade riktlinjer

[WCAG 2.0 - 3.3.2 \(A\)](#)

[WCAG 2.1 - 3.3.2 \(A\)](#)

[Vägledning för webbutveckling - R55 \(1\)](#)

[EN 301 549 - 9.3.3.2](#)

FS60: Ledtexter är knutna till respektive formulärobjekt

✓ Inga fel funna

Bakgrund

Ledtexten är den text som anger för användaren vad denne ska mata in i ett visst fält, vad en kryssruta/radioknapp har för betydelse eller vad användaren ska välja i en lista. Dessa ledtexter är ofta visuellt tydligt placerade nära formulärobjektet (textfältet/kryssrutan/radioknappen/listan/knappen) i fråga vilket gör att användare som ser sidan och kan läsa texten blir tydligt informerade om syftet med formulärobjektet.

För att detta ska fungera för användare som har behov av olika hjälpmedel för att ta till sig informationen måste det finnas en tydlig koppling i koden mellan ledtexten och formulärobjektet. Detta åstadkoms med elementet label. Använd label-element för att lägga in samtliga ledtexter. Label-elementen måste också kopplas korrekt till respektive formulärobjekt med attributet for som ska peka på formulärobjektets id.

Label-elementet ska inte enbart inkludera ledtexten utan också eventuell förtydligande och eventuellt felmeddelande som är knutet till formulärobjektet.

Notera att du bara ska ha 1 label-element per formulärobjekt.

Om gränssnittet kodas med html5 ska formulärfälten vara märkta med vilken typ av fält det rör sig om.

Exempel på formulär:

```
<form action="http://www.funka.com/survey" method="post">

  <label for="namn">Namn</label>
```

```
<input type="text" name="namn" id="namn" />

<label for="country">Land</label>
<select name="selectCountry" id="country">
  <option disabled>Välj</option>
  <option>Norge</option>
  <option>Sverige</option>
</select>

<fieldset>
  <legend class="hidden">Godkänn</legend>
  <label for="yes">
    <input type="radio" name="accept" id="yes">Ja, jag
godkänner</label>
  <label for="no">
    <input type="radio" name="accept" id="no">Nej, jag
godkänner inte</label>
</fieldset>

<label for="comments">Kommentar</label>
<textarea name="comments" id="comments">
</textarea>
<input type="submit" value="Skicka svar" />
</form>
```

Koden genererar följande formulär:

Namn

Land

 ☐

Ja, jag godkänner

☐

Nej, jag godkänner inte

Kommentar

Skicka svar

Kommentar

Ledtexterna läses upp i samband med att fokus fås på inmatningsfältet och punkten bedöms som godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.0 - 2.4.6 \(AA\)](#)

[WCAG 2.0 - 3.3.2 \(A\)](#)

[WCAG 2.0 - 4.1.2 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 2.4.6 \(AA\)](#)

[WCAG 2.1 - 3.3.2 \(A\)](#)

WCAG 2.1 - 4.1.2 (A)

Vägledning för webbutveckling - R55 (1)

EN 301 549 - 9.1.3.1

EN 301 549 - 9.2.4.6

EN 301 549 - 9.3.3.2

EN 301 549 - 9.4.1.2

FS70: Inmatningsfält i formulär går att identifiera genom korrekt uppmärkning i attributet autocomplete

– Inte aktuellt

Bakgrund

Långa och komplexa formulär utgör ofta ett stort problem för många användare, inte minst användare med kognitiva funktionsnedsättningar. Ju enklare formulären är och desto mindre användaren behöver fylla i, desto mindre är risken för problem. Det är många grupper som har extra stora problem med formulär. Förutom användare med kognitiva funktionsnedsättningar utgör de också problem för användare med skrivsvårigheter och användare med grav synnedsättning. Attributet autocomplete gör det möjligt att underlätta för användare att fylla i formulärsfält som ska innehålla information som ofta återkommer, exempelvis namn, e-postadress och telefonnummer. Användaren behöver endast fylla i informationen vid ett tillfälle och kan låta webbläsaren spara informationen till nästa gång som användaren ska fylla i samma information i andra formulärsfält. Autocomplete används på inmatningsfält av olika typer som till exempel text, email, number, password.

```
<form action="/prenumerera.html">
Förnamn:<input type="text" name="fnamn" autocomplete="given-
name" />
Efternamn: <input type="text" name="enamn"
autocomplete="family-name" />
E-post: <input type="email" name="epost" autocomplete="email"
/>
<input type="submit" value="Prenumerera" />
</form>
```

Det finns ett 40-tal möjliga värden att använda i attributet autocomplete. I vissa

situationer är det inte önskvärt att webbläsaren fyller i formulärsfälten. Exempel på sådana situationer är när användaren ska ange en engångskod vid inloggning till en e-tjänst eller engångslösenord som skickas när ett lösenord behöver förnyas. Då kan den automatiska kompletteringen stängas av genom att sätta attributet autocomplete till off på specifika formulärsfält.

Hjälpmedel för personer med kognitiva funktionsnedsättningar kan fånga upp attributet autocomplete i formulärsfält och ge användaren en enklare och mer lättbegriplig beskrivning av den information som formulärsfältet ska innehålla. Ett exempel är när autocomplete sätts till värdet "tel" vilket ersätter eller kompletterar formulärsfältets ledtext med en telefon-ikon.

För att möta kravet på den här punkten ska formulärsfält som finns med i W3Cs lista "Input Purposes for User Interface Components" ha attributet autocomplete satt till de värden som anges i listan.

[Input Purposes for User Interface Components](#)

Kommentar

Vi ser inga eller komplexa formulär i appen.

Kopplade riktlinjer

[WCAG 2.1 - 1.3.5 \(AA\)](#)

[Vägledning för webbutveckling - R52 \(4\)](#)

[EN 301 549 - 9.1.3.5](#)

FS80: Utökade beskrivningar av ett specifikt formulärobject är kopplade till detta formulärobject

– Inte aktuellt

Bakgrund

Det är inte alltid att det räcker med en vanlig ledtext för att förklara för användaren vad, hur och varför viss information i formuläret måste fyllas i. När det uppstår ett behov av en utökad förklaring, exempelvis för att förtydliga datumformat, eller varför du ber användaren att fylla i viss information, ska förklaringen vara kopplad till formulärobjectet lika entydigt som när det gäller den vanliga ledtexten.

Det enklaste sättet är att låta den utökade beskrivningen vara en del av label-elementet. Det är den mest robusta metoden. Du kan också jobba med wai-aria attributen aria-labelledby och aria-describedby men då kommer en del användare att missa informationen. Om det är ett problem eller ej måste redas ut från fall till fall eftersom det beror på situationen och vilka användare systemet riktar sig till.

Kommentar

Inga utökade beskrivningar kan hittas i appen. Punkten bedöms som ej aktuell.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R101 \(3\)](#)

[EN 301 549 - 9.1.3.1](#)

FS100: Viktiga formulär ger användarna möjlighet att läsa en sammanställning av informationen som ska skickas innan de bekräftar och skickar formuläret

– Inte aktuellt

Bakgrund

När en webbaserad funktion används för juridiskt bindande åtaganden eller för att föra över pengar är det extra viktigt att användaren faktiskt förstår vad som kommer att hända. Det är också lätt hänt att råka trycka på "OK" i en webbtjänst utan att vara säker på följderna. För att undvika onödiga problem bör användaren få möjlighet att i ett sista steg läsa igenom en sammanställning och bekräfta händelsen. Detta kan handla om:

- Juridiskt bindande åtaganden
- Ekonomiska transaktioner
- Lagrad information som ändras eller tas bort

Kommentar

Vi hittar inga viktiga formulär i appen.

Kopplade riktlinjer

[WCAG 2.0 - 3.3.4 \(AA\)](#)

[WCAG 2.1 - 3.3.4 \(AA\)](#)

[Vägledning för webbutveckling - R71 \(3\)](#)

[Vägledning för webbutveckling - R77 \(2\)](#)

[EN 301 549 - 9.3.3.4](#)

FS120: Knappars funktion framgår tydligt

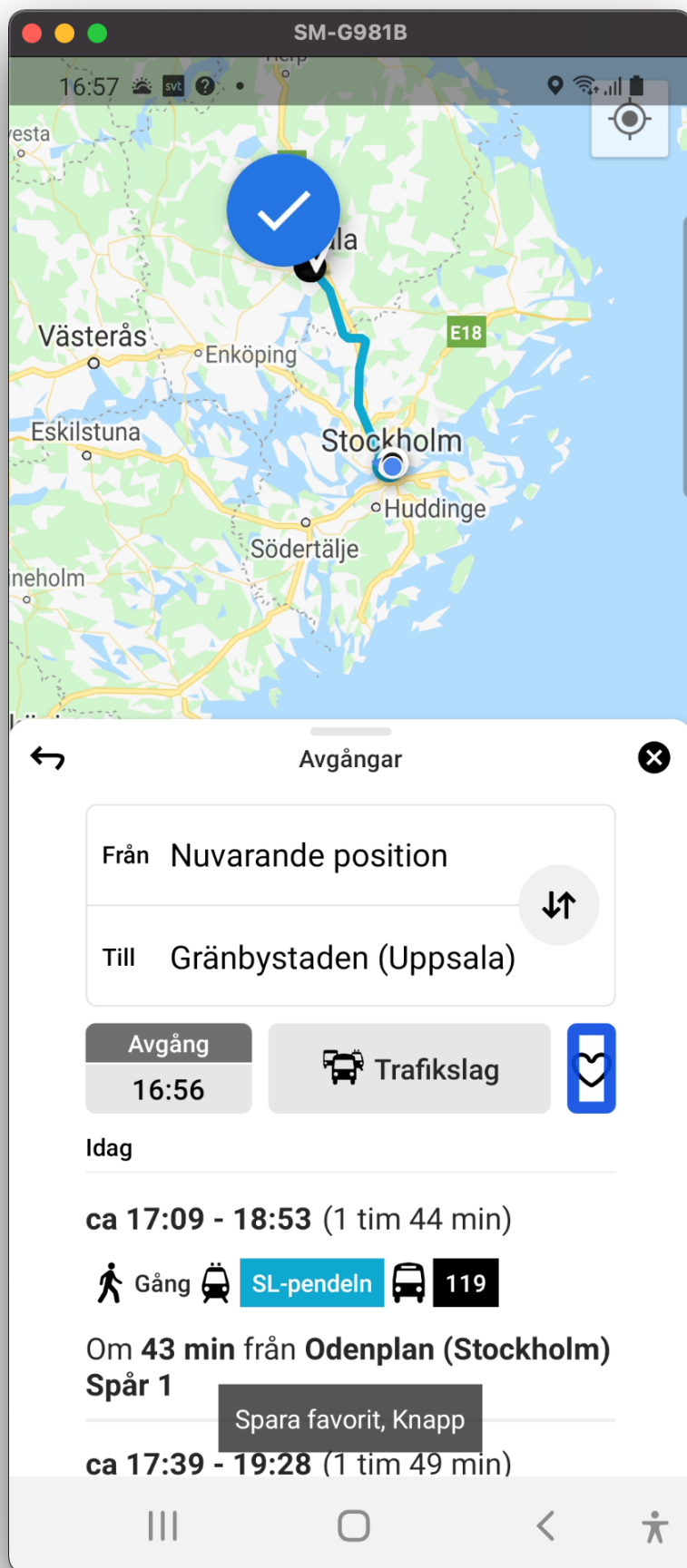
× Underkänd

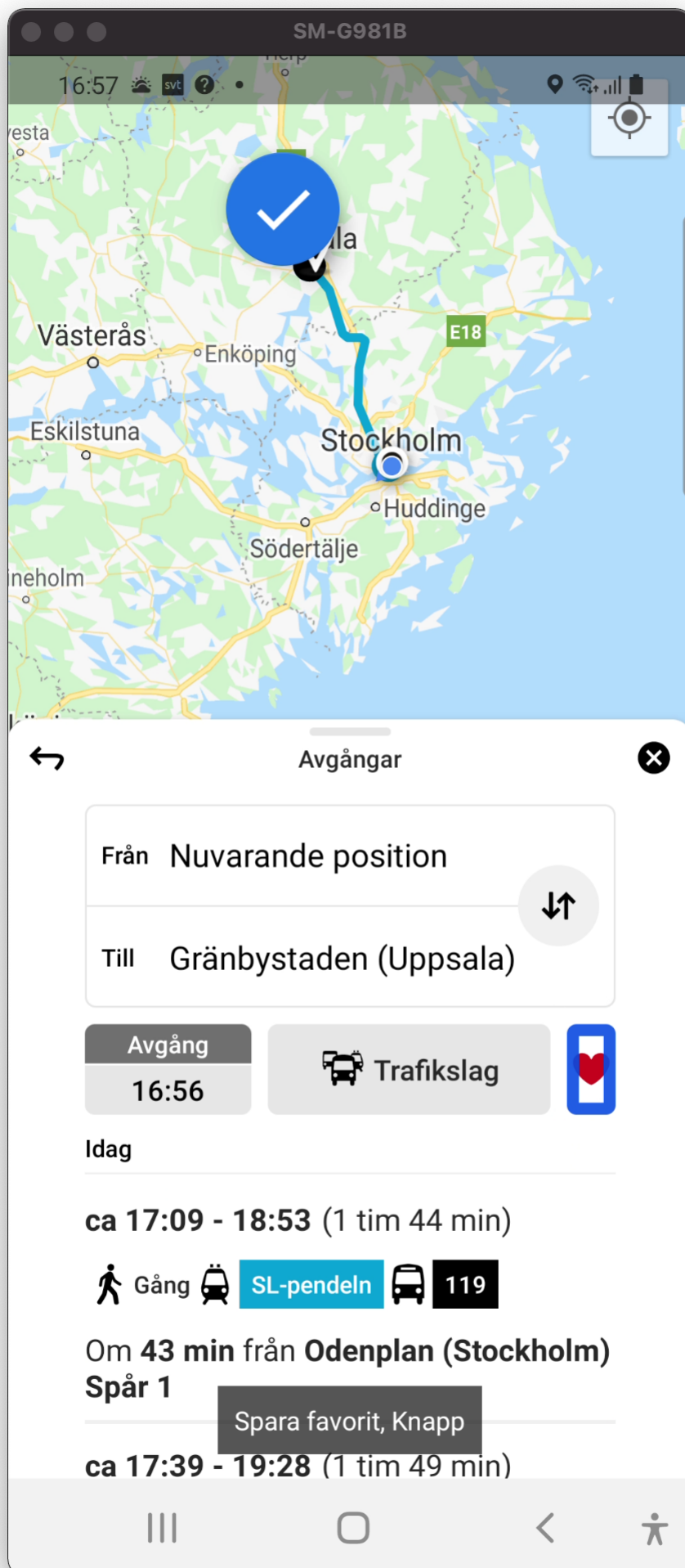
Bakgrund

Det är viktigt att användare förstår vad som kommer att hända när de klickar på en knapp. Knappar bör därför vara självförklarande så att användaren förstår deras funktion. En knapp som heter "OK" är exempelvis inte lika tydlig som en knapp som heter "Kommentera" eller "Prenumerera".

Kommentar

Knappen för att spara en favoritresa har samma titel som en redan favoriserad resa. En skärmläseanvändare kommer att få svårt att veta vilket tillstånd knappen har när det inte framgår i uppläsningen.





Vi har också funnit en knapp som saknar ett namn.



Rekommendationer

- Lägg till strängen "markerad/inte markerad" så att uppläsningen blir: "Spara favoritresa, markerad/inte markerad, knapp".
- Kontrollera att alla klickbara element har ett namn.

Kopplade riktlinjer

[WCAG 2.0 - 1.1.1 \(A\)](#)

[WCAG 2.0 - 4.1.2 \(A\)](#)

[WCAG 2.1 - 1.1.1 \(A\)](#)

[WCAG 2.1 - 4.1.2 \(A\)](#)

[Vägledning för webbutveckling - R60 \(2\)](#)

[EN 301 549 - 9.1.1.1](#)

[EN 301 549 - 9.4.1.2](#)

FS130: Delar i formulären är grupperade

✓ Inga fel funna

Bakgrund

Längre formulär måste delas upp och grupperas. Detta gör det enklare att se vilka delar som hör samman, både visuellt och i ett hjälpmedel. Grupperingar av objekt görs med hjälp av fieldset och legend.

Exempel på ett formulär med grupperingar:

```
<form action="http://funka.com/survey" method="post"
onsubmit="return checkForm(f);">
```

```
<fieldset>
```

```
<legend>Leveransadress</legend>
```

```
<label for="name1">Namn:</label>
```

```
<input type="text" name="name1" id="name1">
```

```
<label for="adress1">Adress:</label>
```

```
<input type="text" name="adress1" id="adress1">
```

```
<label for="pnum1">Postnummer:</label>
```

```
<input type="text" name="pnum1" id="pnum1">
```

```
<label for="ort1">Ort:</label>
<input type="text" name="ort1" id="ort1">
</fieldset>

<fieldset>

  <legend>Faktureringsadress</legend>

  <label for="name2">Namn:</label>
  <input type="text" name="name2" id="name2">

  <label for="adress2">Adress:</label>
  <input type="text" name="adress2" id="adress2">

  <label for="pnum2">Postnummer:</label>
  <input type="text" name="pnum2" id="pnum2">

  <label for="ort2">Ort:</label>
  <input type="text" name="ort2" id="ort2">

</fieldset>

<input type="submit">

</form>
```

Så fort du har en grupp med radioknappar eller kryssrutor måste du använda fieldset och legend för att ge gruppen en gemensam beskrivning. Utan detta kan man lite förenklat säga att det är som att uppmana användaren att svara på en fråga utan att denne får höra frågan.

Kommentar

Växlingsknappar är grupperade, inga fel funna.

Kopplade riktlinjer

WCAG 2.0 - 1.3.1 (A)

FS140: När fel uppstår meddelas det tydligt för användaren

× Underkänd

Bakgrund

Om det finns en möjlighet att göra fel kommer vissa användare att göra fel. Det är då viktigt att användaren får veta att något blivit fel.

Ge ett samlat felmeddelande överst

Det finns flera olika sätt att visa att ett fel har uppstått men för att så många användare som möjligt ska uppfatta det ska de flesta formulär erbjuda ett samlat felmeddelande överst. Detta ska tydligt ange:

- Att det blivit fel. Använd gärna både en rubrik och en ikon för att ange att det är ett felmeddelande.
- Vilka fel som inträffat. Gärna i form av en punktlista.
- Var felen inträffat. Namnge formulärsobjekten där felen finns.
- Om möjligt även vad användaren ska göra för att korrigera felen.

Beskriv felet vid respektive fält

På de platser i formuläret där det blivit fel behöver det finnas ett tydligt felmeddelande, gärna med en ikon. Även här ska felet beskrivas för användaren.

Vad kan jag göra mer?

Ännu bättre blir det om du även erbjuder livevalidering så att fel upptäcks under tiden som användaren fyller i fälten. Tänk dock på att inte alla användare kommer att se detta, det måste därför gå att skicka in formuläret även utan att användaren korrigerat felen. När användaren skickar in ska då ett samlat felmeddelande enligt ovan ges.

I det samlade felmeddelandet kan du med fördel också länka ner till de enskilda felen i formuläret, så att användaren slipper leta.

Andra typer av fel

Ett annat exempel på fel som kan uppstå är när användaren försöker ladda en sida som inte finns. Då måste webbplatsen ge ett tydligt felmeddelande som

anger att sidan i fråga inte finns och ge användaren möjlighet att hitta informationen på annat sätt.

Kommentar

Vi har inte hittat ett konsekvent beteende när det blir fel. Ibland visas ett dialogfönster som meddelar att något inte är korrekt, exempelvis när saldot i reskassan inte räcker till, när en värdekod fylls i inkorrekt eller när personnumret inte hittas vid en sjukresa. Vi har också hittat ett felmeddelande som endast är i färg.

I exemplet nedan behöver felet också skrivas ut i textform för att det skall vara godkänt, det räcker således inte att markera ledtexten med färg.



Rekommendationer

- Säkerställ att vid inmatningsfel visas ett felmeddelande även i textform. Se bakgrundstexten för exempel på lösningar.

Kopplade riktlinjer

[WCAG 2.0 - 3.3.1 \(A\)](#)

[WCAG 2.1 - 3.3.1 \(A\)](#)

[Vägledning för webbutveckling - R2 \(1\)](#)

[EN 301 549 - 9.3.3.1](#)

FS150: Felmeddelanden knyts till respektive formulärsubjekt

× Underkänd

Bakgrund

När felmeddelanden läggs in i formuläret och avser ett specifikt formulärsubjekt eller en grupp med objekt (exempelvis en grupp med radioknappar) så måste det knytas till formulärsubjektet eller gruppen. Det görs normalt sett genom att lägga in felmeddelandet som en del av label-elementet eller legend-texten.

Eventuellt kan det innebära att felmeddelandet läggs in två gånger, en gång som visas visuellt och en som ligger dold i den befintliga ledtexten. I det läget är det bra att dölja det visuella felmeddelandet för hjälpmedel och se till att det visuellt dolda felmeddelandet läses. Det kan exempelvis göras så här:

Html-kod:

```
<label for="namn">Namn: <span class="dold">Fel: Du har missat  
att skriva in ditt namn</span></label>  
  
<input id="namn" type="text">  
  
<span aria-hidden="true">Du har missat att skriva ditt  
namn</span>
```

Css-kod:

```
.dold { position: absolute; left: -20000px;}
```

Givetvis är det bästa om du bara behöver lägga in felmeddelandet en gång och göra det synligt för alla.

Kommentar

Detta felmeddelande visas endast visuellt. Här behöver det läsas upp för en skärmläseanvändare när fokus är kvar i inmatningsfältet att det angivna beloppet är för lågt. Punkten blir därför underkänd.

Rekommendationer

- Märk upp felmeddelandet med attributet nedan för att TalkBack skall läsa upp felmeddelandet:
- `announceForAccessibility("felmeddelande")`

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

FS170: Alla fel beskrivs med text

✕ Underkänd

Bakgrund

Felmeddelanden kan tänkas innehålla bilder eller till och med filmer, men det är viktigt att det alltid finns ett felmeddelande i form av text. Det räcker alltså inte med en ikon för att markera att något är fel, det ska alltid uttryckligen stå i text vad som är fel.

Kommentar

Se punkten (FS140). Där visas ett fel som endast är markerat i färg. Även om ledtexten är i text behöver felmeddelandet vara skrivet i text också.

Rekommendationer

- Beskriv vad som är fel och skriv ut det i text.

Kopplade riktlinjer

[WCAG 2.0 - 3.3.1 \(A\)](#)

[WCAG 2.1 - 3.3.1 \(A\)](#)

[EN 301 549 - 9.3.3.1](#)

FS180: När det är möjligt ska det framgå hur felen kan åtgärdas

! Inga fel funna men kan förbättras

Bakgrund

Beskriv så detaljerat som möjligt vad som är fel och hur användaren kan åtgärda det. Exempelvis kan du ge förslag på vad som kan tänkas vara fel, exempel:

Du måste ange en e-postadress som innehåller ett @-tecken

Istället för

Du måste ange en giltig e-postadress

Kommentar

För felmeddelandet som visas i FS140 saknas ett riktigt felmeddelande i text och en beskrivning på hur det skall åtgärdas. Ledtexten är dock beskrivande nog att användaren får den information de behöver. Den förbättring vi ser är att ha ett dedikerat felmeddelande med en tydligare beskrivning på vad som är fel.

Rekommendationer

- Skriv ut ett felmeddelande i stil med: "Det angivna beloppet är för låg, ange ett belopp mellan 100 - 1000 kr."

Kopplade riktlinjer

[WCAG 2.0 - 3.3.3 \(AA\)](#)

[WCAG 2.1 - 3.3.3 \(AA\)](#)

[Vägledning för webbutveckling - R2 \(1\)](#)

[Vägledning för webbutveckling - R149 \(1\)](#)

[EN 301 549 - 9.3.3.3](#)

Tabeller

TB40: Tabelldata är kodad med korrekt tabellkod

– Inte aktuellt

Bakgrund

Koda i första hand alltid tabeller med korrekt tabellkod. Det går idag att skapa tabeller som ser ut som tabeller men som inte är kodade som korrekta tabeller. Det finns också html5-roller och wai-aria attribut som kan användas för att skapa tabeller i stället för traditionell html-kod, men det är lösningar som idag inte fungerar alltid lika bra för användare med hjälpmedel. Tabeller ska därför alltid helst byggas upp med elementen table, tr, td och th.

Kommentar

Inga tabeller hittas i appen.

Kopplade riktlinjer

[WCAG 2.0 - 4.1.2 \(A\)](#)

[WCAG 2.1 - 4.1.2 \(A\)](#)

[EN 301 549 - 9.4.1.2](#)

TB50: Rubriker till tabeller är kodade med elementet caption

– Inte aktuellt

Bakgrund

Tabeller är en bra teknik som finns för att presentera data. För att en användare ska ha en möjlighet att förstå tabellen måste det finnas tydliga tabellrubriker som lagts in på ett korrekt sätt.

Om det inte finns särskilda skäl ska elementet caption användas för att lägga in en rubrik till tabellen. Denna rubrik ska kortfattat förklara vad tabellen innehåller för information. Caption-elementet läggs överst i tabellen på samma nivå som tabellrader (tr-element):

```
<table>
  <caption>Flygpriser Stockholm-Luleå</caption>
  <tr>
    ...
```

Kommentar

Inga tabeller hittas i appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.1.3.1](#)

TB60: Rad- och kolumnrubriker kodas med elementet th

– Inte aktuellt

Bakgrund

För att användarna ska kunna tolka innehållet i tabellerna korrekt är det viktigt att tabellerna har tydliga kolumn- och radrubriker. Dessa rubriker ska inte bara vara visuellt tydliga, de måste också vara strukturellt rätt gjorda med elementet th. Th talar om att cellen är en "Table Header", alltså en tabellrubrik, till skillnad från elementet td som anger att cellen är en "Table Data cell", alltså en datacell. Alla tabellrubriker ska läggas in med elementet th.

Kommentar

Inga tabeller hittas i appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R98 \(4\)](#)

[EN 301 549 - 9.1.3.1](#)

TB70: Attributet scope används för att ange riktning för rad- och kolumnrubriker när det behövs

– Inte aktuellt

Bakgrund

Tabellrubrikerna ska även knytas korrekt med de dataceller som är underordnade rubriken. För enkla tabeller med en axel räcker det om tabellrubriken är identifierad som <th>. I tabeller med fler axlar ska attributet scope anges om rubriken avser kolumn (värdet "col") eller rad (värdet "row"). För tabeller i mer än två axlar och andra rubrikformer krävs en mer avancerad uppmärkning.

Exempel på tabell med rad och kolumnrubriker:

```
<table>
  <tr>
    <td></td>
    <th scope="col">Frukter</th>
    <th scope="col">Grönsaker</th>
    <th scope="col">Bär</th>
  </tr>

  <tr>
    <th scope="row">Röda</th>
    <td>Persikor</td>
    <td>Rödbetor</td>
    <td>Jordgubbar</td>
  </tr>

  <tr>
    <th scope="row">Gröna</th>
    <td>Äpplen</td>
    <td>Isbergssallad</td>
    <td>Krusbär</td>
  </tr>

  <tr>
    <th scope="row">Gula</th>
    <td>Bananer</td>
```

```

        <td>Paprika</td>
        <td>Hjortron</td>
    </tr>

    <tr>
        <th scope="row">Blå</th>
        <td>Plommon</td>
        <td>Aubergine</td>
        <td>Blåbär</td>
    </tr>
</table>

```

;

	Fruktar	Grönsaker	Bär
Röda	Persikor	Rödbetor	Jordgubbar
Gröna	Äpplen	Isbergssallad	Krusbär
Gula	Bananer	Paprika	Hjortron
Blå	Plommon	Aubergine	Blåbär

Kommentar

Inga tabeller hittas i appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.1.3.1](#)

TB100: Komplexa tabeller kompletteras med relevant kod

– Inte aktuellt

Bakgrund

Komplexa tabeller måste ha mer specifika kopplingar mellan rubriker och dataceller än enkla tabeller. Detta kan åstadkommas genom att på varje rubrik ange ett unikt id och att sedan i de olika cellerna referera till cellens rubriker i attributet headers.

I komplexa tabeller kan även logiska kategorier skapas med hjälp av attributet axis som används för att gruppera rubriker i gemensamma kategorier.

Exempel på en mer komplex tabell där rubrikcellerna och datacellerna kopplats samman:

```
<table>
  <tr>
    <th></th>
    <th colspan="2" id="birds">Fåglar</th>
    <th colspan="2" id="fish">Fiskar</th>
  </tr>

  <tr>
    <th></th>
    <th id="d071223a" headers="birds">Svenska</th>
    <th id="d071224a" headers="birds">Afrikanska</th>
    <th id="d071223b" headers="fish">Svenska</th>
    <th id="d071224b" headers="fish">Afrikanska</th>
  </tr>

  <tr>
    <th id="oneColor">Enfärgade</th>
    <td headers="birds d071223a oneColor">Kråka</td>
    <td headers="birds d071224a oneColor">Svart drongo</td>
    <td headers="fish d071223b oneColor">Sill</td>
    <td headers="fish d071224b oneColor">Guldanthias</td>
  </tr>
```

```

<tr>
  <th id="striped">Randiga</th>
  <td headers="birds d071223a striped">Grågäss</td>
  <td headers="birds d071224a striped">Härfågel</td>
  <td headers="fish d071223b striped">Aborre</td>
  <td headers="fish d071224b striped">Äkta kejsarfisk</td>
</tr>

<tr>
  <th id="dotted">Prickiga</th>
  <td headers="birds d071223a dotted">Jorduggla</td>
  <td headers="birds d071224a dotted">Rödgul barbett</td>
  <td headers="fish d071223b dotted">Bäckröding</td>
  <td headers="fish d071224b dotted">Muräna</td>
</tr>

</table>

```

	Fåglar		Fiskar	
	Svenska	Afrikanska	Svenska	Afrikanska
Enfärgade	Kråka	Svart drongo	Sill	Guldanthias
Randiga	Grågäss	Härfågel	Abborre	Äkta kejsarfisk
Prickiga	Jorduggla	Rödgul barbett	Bäckröding	Muräna

Om du har en så här pass komplex tabell bör du i första hand värdera att dela upp den i flera, enklare tabeller eller att på annat sätt förenkla uppställningen så att du klarar dig med enbart scope för att associera rubriker och dataceller.

Kommentar

Inga tabeller hittas i appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[Vägledning för webbutveckling - R98 \(4\)](#)

Script & wai-aria

SW20: Använd wai-aria för att förmedla information som inte kan förmedlas med html och som ökar användarnyttan

× Underkänd

Bakgrund

Standarden wai-aria innehåller en uppsättning attribut som kan användas för att förtydliga olika html-objekts funktion, värde och roll. En del av det som ingår i wai-aria är sådant som det finns bra lösningar för i html5, då ska html-lösningen användas, men mycket i wai-aria är sådant som tydliggör och ökar tillgängligheten ytterligare utöver det som är inbyggt i html5.

Ett exempel där html5 och wai-aria överlappar är navigering. Du ska i första hand använda html5 attributet <nav> för att visa att något är en meny, men det finns en möjlighet i wai-aria att ange det här också genom attributet role="navigation". Vår rekommendation är alltså att i första hand använda lösningen som finns i html5.

Ett tydligt exempel på hur wai-aria ökar användarnyttan är attributet aria-expanded. Det här används på knappar som faller ut och faller ihop områden eller ytor, exempelvis ska meny-knappen för mobilmenyn ha aria-expanded. Genom att ange true för när menyn är utfälld och false för när menyn inte är utfälld kan du ge en indikation till användare som inte ser gränssnittet om menyn är utfälld eller inte.

Överanvänd inte wai-aria

Även om wai-aria innehåller mycket som är bra, så är det viktigt att inte överanvända standarden. Du ska inte bara lägga till objekt om du inte vet att det ökar användarnyttan. Undvik att lägga på wai-aria om html-koden redan tydliggör funktionen, exempelvis är det onödigt att skriva <button role="button">, det riskerar bara att skapa dubbel uppläsning för användarna. Använd inte heller role="menu" och role="menuitem" för vanliga webbplatsmenyer, det här är tänkt för applikationer, exempelvis texteditorer, och användarna blir bara störda över att höra hjälpmedlen läsa "menyobjekt" för varje länk i menyn.

Kommentar

WAI-ARIA finns inte direkt-översatt till native-appar. Denna punkt har vi därför bedömt utifrån vad som är möjligt att göra i appar som skulle underlätta för

hjälpmedelsanvändare.

Android-appen står sig väl i denna punkt. Antalet sökträffar läses upp, likaså den återstående biljettiden.

Det vi saknar för att bedömningen blir godkänd är att samtliga felmeddelanden läses upp. Se punkt (FS140).

Kopplade riktlinjer

WCAG 2.0 - 4.1.2 (A)

WCAG 2.1 - 4.1.2 (A)

EN 301 549 - 9.4.1.2

SW50: När nya element läggs in i sidan placeras de på rätt plats både visuellt och strukturellt

✓ Inga fel funna

Bakgrund

Ibland läggs nytt innehåll eller nya funktioner in i sidan utan att den laddas om. När det sker är det viktigt att den nya funktionen eller informationen uppträder på rätt plats både visuellt och strukturellt. Om ett sökresultat dyker upp i ett nytt område på skärmen när användaren börjar skriva i sökfältet är det viktigt att detta område är placerat visuellt tydligt där det hör hemma, men också på rätt plats i strukturen så att hjälpmedlen strukturellt presenterar informationen på rätt plats. Det bästa är om detta görs så att den nya koden injiceras på rätt plats, exempel:

```
<label for="sok">Sök:</label>
<input type="search" id="sok" />
<!-- Sökförslag läses in här -->
<input type="submit" value="Sök" />
```

Det finns andra sätt att lösa detta på också, exempelvis genom att med skript känna av vad användaren gör och förflytta fokus så att effekten för användaren blir den samma. Hur det fungerar bör alltid testas praktiskt med de vanligaste hjälpmedlen.

Ett vanligt problem är också att ett nytt område visuellt placeras korrekt men hamnar sist på sidan i strukturen. Om ett hjälpmedel inte hittar något förrän efter sidfoten är det stor risk att användaren aldrig kommer att hitta det.

Kommentar

Element laddas in där de förväntas, punkten godkänns.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.2 \(A\)](#)

[WCAG 2.0 - 2.4.3 \(A\)](#)

[WCAG 2.1 - 1.3.2 \(A\)](#)

[WCAG 2.1 - 2.4.3 \(A\)](#)

[Vägledning för webbutveckling - R122 \(1\)](#)

[Vägledning för webbutveckling - R136 \(1\)](#)

[EN 301 549 - 9.1.3.2](#)

[EN 301 549 - 9.2.4.3](#)

SW60: När nytt innehåll eller nya områden öppnas får användaren direkt fokus på detta

✓ Inga fel funna

Bakgrund

Om ett täckande lager, eller ett nytt område öppnas dynamiskt på sidan är det viktigt att användarna direkt får fokus på detta (om det inte stör användaren).

Exempel på undantag är sökförslag. Användarens fokus ska inte flyttas automatiskt till sökförslagen så fort de dyker upp eftersom det skulle störa.

För en användare som navigerar med tangentbordet är det viktigt att direkt komma in i det nya lagret eller innehållet utan att behöva tabba till slutet av sidan.

Om det är ett formulär kan fokus sättas i det första fältet, annars ska första tabbsteget efter länken som öppnade lagret eller resulterade i att det nya området lästes in, leda in till första länken i området.

Tänk också på att läsordningen måste bli korrekt för användare med uppläsande hjälpmedel. När de fortsätter läsningen nedåt på sidan från den länk som öppnade lagret ska läsningen fortsätta inne i lagret i fråga. Detta kan kräva att du testat det med hjälpmedel.

Om det gäller meddelanden och dialogrutor bör wai-aria:s roller för meddelanden och meddelandedialoger användas. Om det är ett formulär bör det vara rollen alerdialog:

```
role="alerdialog"
```

Annars rollen alert:

role="alert"

[Information om wai-aria \(W3C:s webbplats\)](#)

Kommentar

Fokus fås när nya områden eller fönster öppnas och punkten blir godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.3 \(A\)](#)

[WCAG 2.1 - 2.4.3 \(A\)](#)

[Vägledning för webbutveckling - R136 \(1\)](#)

[EN 301 549 - 9.2.4.3](#)

SW70: När status ändras informeras hjälpmedelsanvändare utan att fokusmarkeringen flyttas

✕ Underkänd

Bakgrund

Uppdateringar i gränssnittet som sker utan att sidan laddas om är särskilt svåra för användare med gravt nedsatt syn att uppfatta. Det kan vara att det sker en förändring i en pågående process, exempelvis en uppdaterad köplats på en webbplats för försäljning av konsertbiljetter. Det kan också handla om att informera användaren om hur många sökträffar som en sökning har resulterat i, eller validering av fel i ett formulär. Det är problematiskt för användaren att snabbt förflytta sig till en annan del av gränssnittet för att hinna läsa ett statusmeddelande innan det är inaktuellt eller har uppdaterats.

Exempel på statusändringar är:

- Information om antalet sökträffar i en sökning
- En förändring sker i en väntande applikation
- En förändring sker i en pågående process
- Fel uppstår i ett formulär

När ett statusmeddelande visas på skärmen ska användaren få information om det utan att behöva flytta sig till ett annat område på sidan. Attributen `role="alert"`, `aria-live` och `aria-busy` kan användas för att informera skärmläsaranvändare om statusmeddelanden som visas på andra delar av sidan än där användaren för närvarande befinner sig. Detta innebär att användaren får statusmeddelanden, utan att själv behöva leta efter förändringar eller att uppdatera sidan. Hjälpmedel

för användare med kognitiva funktionsnedsättningar kan förtydliga, fördröja eller dämpa statusmeddelanden, beroende på vad användaren föredrar.

Sök

Din sökning på tillgänglighet resulterade i 538 träffar

→ Nya upphandlingskrav för tillgänglighet 2017

Den europeiska standarden för krav på tillgänglighet vid offentlig upphandling av IT-varor och tjänster ... EN 301 549, ligger till grund för de krav på tillgänglighet som gäller i Sverige från och med 1 januari

```
<div class="medium-12 columns">
<h2 aria-live="assertive" aria-atomic="true">
Din sökning på
<strong><mark>tillgänglighet</mark></strong>resulterade i 538
träffar
</h2>
</div>
```

Wai-aria-attribut kan användas i kombination för att göra det tydligare för användare att en statusuppdatering är särskilt viktig. Ett exempel på det är att använda attributen `role="alert"` och `aria-live="assertive"` i meddelanden som informerar användaren om att denne håller på att loggas ut.

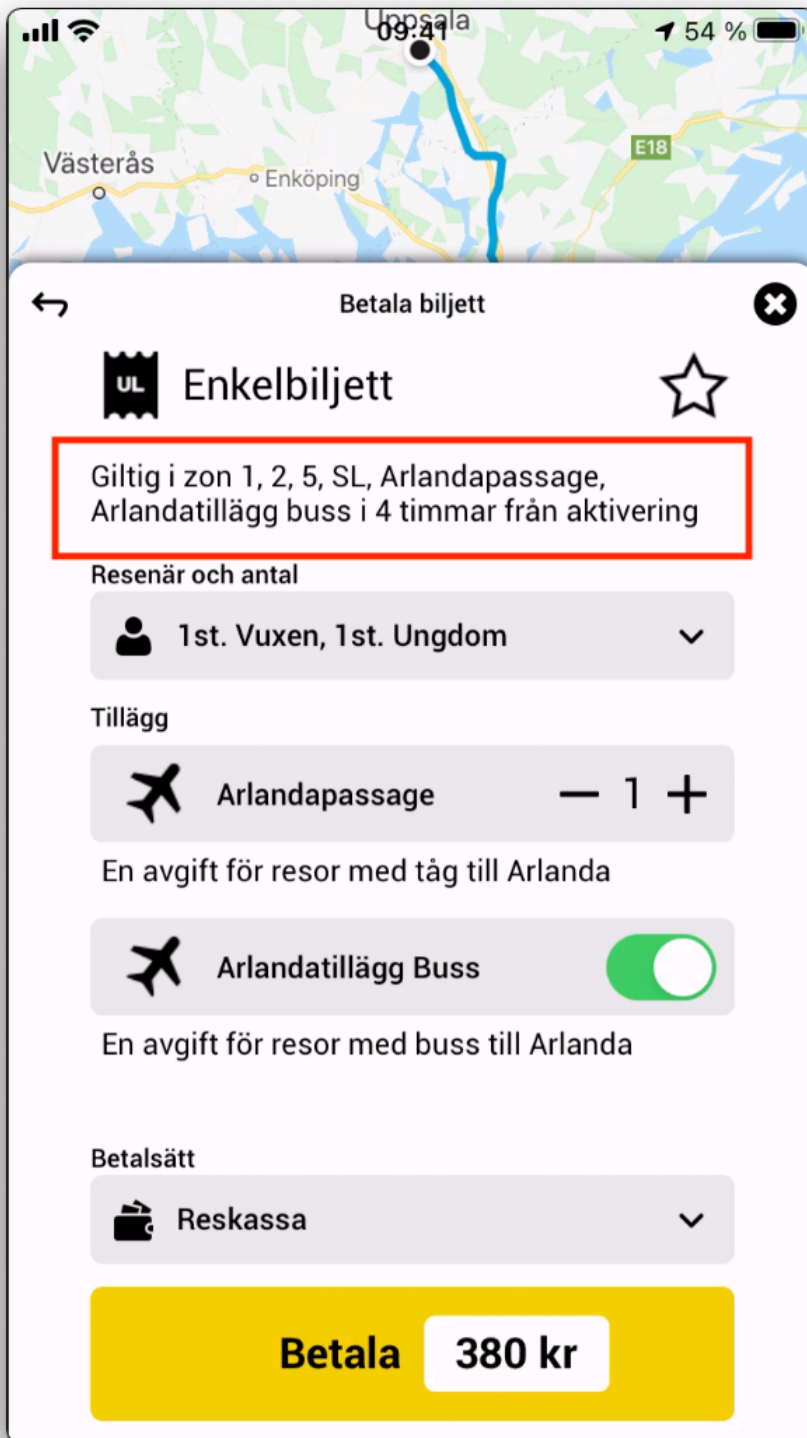
Attributet `aria-live` har tre lägen som bestämmer hur hög prioritet en uppdatering i ett område ska ha:

- **Assertive** = Ett viktigt meddelande som ska avbryta hjälpmedlet så snart som möjligt.
- **Polite** = Ett mindre viktigt meddelande som ska presenteras om användaren gör en paus.
- **Off** = Ett meddelande som inte behöver presenteras för användaren.

Denna riktlinje gäller inte felmeddelanden i modala fönster, innehåll som visas eller döljs med utfällbara objekt eller innehåll som kräver inmatning från användaren. Information om cookies räknas alltså inte som ett statusmeddelande eftersom det ofta kräver att användaren bekräftar sitt samtycke.

Kommentar

Sökförslag och den återstående biljettiden läses upp för en skärmläseanvändare, vilket är jättebra. Det vi saknar är att texten i en köp-process av en biljett också läses upp i samband med att en resenär gör ändringar.



Bilden är lånad från iOS.

Rekommendationer

- Märk upp så att texten blir uppläst efter att en användare gör någon ändring i sin biljett, alternativt att texten flyttas ned i innehållet så att användaren får fokus på texten när de skall trycka på "Betala".

Kopplade riktlinjer

WCAG 2.1 - 4.1.3 (AA)

EN 301 549 - 9.4.1.3

SW80: Innehåll som visas när ett objekt får fokus är fullt användbart och nåbart tills användaren flyttar fokus

✓ Inga fel funna

Bakgrund

Användare med förstorande hjälpmedel ser bara en liten del av skärmen i taget, vilket gör det svårt att överblicka stora områden. Det är även problematiskt för användare att hitta innehåll som ligger utanför det område som för närvarande förstoras. Användare som navigerar med mus eller tangentbord och förstorande hjälpmedel, som följer musmarkörens och tangentbordets rörelser, måste därför kunna flytta fokus utan att ett område försvinner. Denna riktlinje gäller både vid hover med musmarkör och fokus med tangentbordet.

Ett exempel på det här är megamenyer, alltså menyer som består av flera nivåer och presenteras i stora områden som visas när användaren hovrar över menyn. De täcker ofta annat innehåll och försvinner när användaren musmarkör går utanför området. För att användare med förstorande hjälpmedel ska kunna komma åt allt innehåll ska användaren kunna flytta fokus fritt inom området utan att det stängs. Användaren ska också ha möjlighet att stänga området på ett enkelt sätt, exempelvis med escape-tangenten.

Områden som inte skymmer annat innehåll behöver inte ha en stängningsmekanism. Ett exempel är menyer som istället för att öppnas i ett täckande lager, öppnas ovanför sidans innehåll som flyttas ner och placeras nedanför menyn. Inte heller felmeddelanden som beskriver inmatningsfel behöver ha en stängningsmekanism.

Denna riktlinje gäller inte modala fönster eller de tooltips som läggs in med attributet title eftersom dessa implementeras av själva webbläsaren.

Kommentar

Vi ser inget oväntat beteende och punkten bedöms som godkänd.

Kopplade riktlinjer

[WCAG 2.1 - 1.4.13 \(AA\)](#)

[EN 301 549 - 9.1.4.13](#)

SW90: När ett modalt fönster visas ska fokus stanna i modalen tills den är stängd

× Underkänd

Bakgrund

Om du använder modala fönster i ditt gränssnitt, är det viktigt att alla användare kan komma tillbaka till huvudinnehållet på ett enkelt sätt, utan att fastna i det modala fönstret.

En vanlig fel vid användning av modal fönster är att användaren med hjälp av tangentbord eller hjälpmedel kan hamna bakom fönstret trots att den inte är stängd. Med andra ord är det ett krav att användare som navigerar med tangentbord eller hjälpmedel bara kan orientera sig innanför fönstret tills att användaren har stängt det.

När det gäller att spärra hjälpmedel från bakgrunden rekommenderar vi att använda attributet `aria-hidden="true"`, som du kan lägga så högt i strukturen som möjligt för att omfatta allt innehåll som inte ligger i modalfönstret.

Det är däremot mer krävande att spärra bakgrunden för navigering med tangentbord, eftersom du måste upphäva standardbeteendet för varje enskilt objekt, alltså sätta `tabindex="-1"` på alla länkar, knappar och formulärobject, och om det är aktuellt även på andra typer av objekt som redan har `tabindex`-attributet.

Kommentar

Med tangentbordsstyrning kan fokus hamna utanför en modal, se punkt NL30.

Rekommendationer

- Säkra att fokus inte lämnar ett modalt fönster.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.3 \(A\)](#)

[WCAG 2.1 - 2.4.3 \(A\)](#)

[EN 301 549 - 9.2.4.3](#)

SW100: Använd aria-expanded för områden som kan fällas ut och ihop

✓ Inga fel funna

Bakgrund

Attributet aria-expanded ska användas på områden som kan fällas ut och fällas ihop. Attributet anger om området är utfällt eller inte. Det här kan gälla en FAQ lista där svaren fälls ut när användaren klickar på frågan, eller på en meny som dynamiskt fäller ut nästa nivå. Aria-expanded ska bara användas om detta sker dynamiskt, alltså inte när det sker med en sidomladdning.

```
<h3><a aria-controls="svar1" aria-expanded="true"
href="#svar1" onclick="javascript:expand-svar1();"
role="button">När ska jag använda aria-expanded?</a></h3>

<p id="svar1">Du ska använda det när ett område kan fällas ut
och ihop dynamiskt utan sidomladdning.</p>

<h3><a aria-controls="svar2" aria-expanded="false"
href="#svar2" onclick="javascript:expand-svar2();"
role="button">En annan fråga?</a></h3>

<p id="svar2">Ett annat svar.</p>
```

Kommentar

Knapp för att fälla ut hållplatser som passeras är uppmärkt korrekt. Punkten blir godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 4.1.2 \(A\)](#)

[WCAG 2.1 - 4.1.2 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.4.1.2](#)

Automatiska händelser

AE20: Nya fönster och täckande lager öppnas inte utan att användaren valt att göra det

✓ Inga fel funna

Bakgrund

Nya fönster och lager som delvis täcker den egentliga webbsidan skapar problem för flera grupper av användare. De skapar förvirring och är många gånger svåra att stänga. Därför ska användaren alltid få information om att ett nytt fönster eller täckande lager kommer att öppnas. Användaren bör också om möjligt få välja om ett nytt fönster eller lager ska öppnas.

Den här punkten är inriktad på automatiskt öppnade lager och fönster, exempelvis webbplatsundersökningar som lägger sig ovanför det vanliga fönstret när besökaren visar en sida, eller reklamfönster som öppnas när användaren kommer in på webbplatsen. Vanliga länkar som öppnar nya fönster ska meddela detta till användaren, men bedömningen av den informationen görs under punkt NL240 under rubriken Navigation & Länkar.

Normalt sett ska inte nya fönster och täckande lager dyka upp utan att användaren aktivt klickat på en länk som tydligt anger vad som kommer att hända, men ett alternativ kan vara att låta användaren göra en inställning där denne accepterar att den här typen av händelser inträffar.

Ett undantag från denna punkt är datumväljare, förutsatt att användarens fokus är kvar i inmatningsfältet som datumväljaren är kopplad till.

Kommentar

Inga oönskade fönster öppnas, punkten bedöms som godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 3.2.1 \(A\)](#)

[WCAG 2.0 - 3.2.2 \(A\)](#)

[WCAG 2.0 - 3.2.5 \(AAA\)](#)

[WCAG 2.1 - 3.2.1 \(A\)](#)

[WCAG 2.1 - 3.2.2 \(A\)](#)

[WCAG 2.1 - 3.2.5 \(AAA\)](#)

[Vägledning för webbutveckling - R143 \(1\)](#)

[EN 301 549 - 9.3.2.1](#)

[EN 301 549 - 9.3.2.2](#)

AE60: Om automatiska händelser eller tidsgränser används finns det en möjlighet att pausa eller förlänga tidsintervallet

✓ Inga fel funna

Bakgrund

Varje tidsgräns måste vara möjlig att antingen:

- Inaktiveras
- Förlängas. Användaren kan själv styra hur lång tidsgränsen ska vara och maxvärdet är minst tio gånger längre än grundvärdet
- Utökas. Exempelvis genom att en varning kommer upp några minuter innan tidsgränsen löper ut och användaren ges möjlighet att förlänga

Användaren kan exempelvis få möjlighet att inaktivera, utöka eller förlänga tidsgränsen genom en knapp som visas i anslutning till tidsgränsen. Ett annat alternativ är att visa ett meddelande någon minut innan utloggning där användaren kan välja att förlänga inloggningstiden.

Tidsgränsen ska alltid förhålla sig till senaste användaraktivitet som systemet kan känna av, inte till inloggningstidpunkten. Om användaren exempelvis går från steg 1 till steg 2 i en tjänst så ska det nollställa nedräkningen till utloggningen.

Om tidsgränsen är sådan att meningen med funktionen helt går förlorad om denna skjuts upp eller ändras kan det vara ett acceptabelt undantag. Likaså om tidsgränsen är på minst 20 timmar.

Kommentar

Tidsgränser används på biljetterna men det är en skälig anledning och punkten bedöms som inga fel funna

Kopplade riktlinjer

[WCAG 2.0 - 2.2.1 \(A\)](#)

[WCAG 2.0 - 2.2.2 \(A\)](#)

[WCAG 2.0 - 2.2.4 \(AAA\)](#)

[WCAG 2.0 - 3.2.5 \(AAA\)](#)

[WCAG 2.1 - 2.2.1 \(A\)](#)

[WCAG 2.1 - 2.2.2 \(A\)](#)

[WCAG 2.1 - 2.2.4 \(AAA\)](#)

[WCAG 2.1 - 3.2.5 \(AAA\)](#)

[Vägledning för webbutveckling - R131 \(1\)](#)

AE80: Om ett område uppdateras utan att sidan laddas om ska detta vara uppmärkt med WAI-ARIA

× Underkänd

Bakgrund

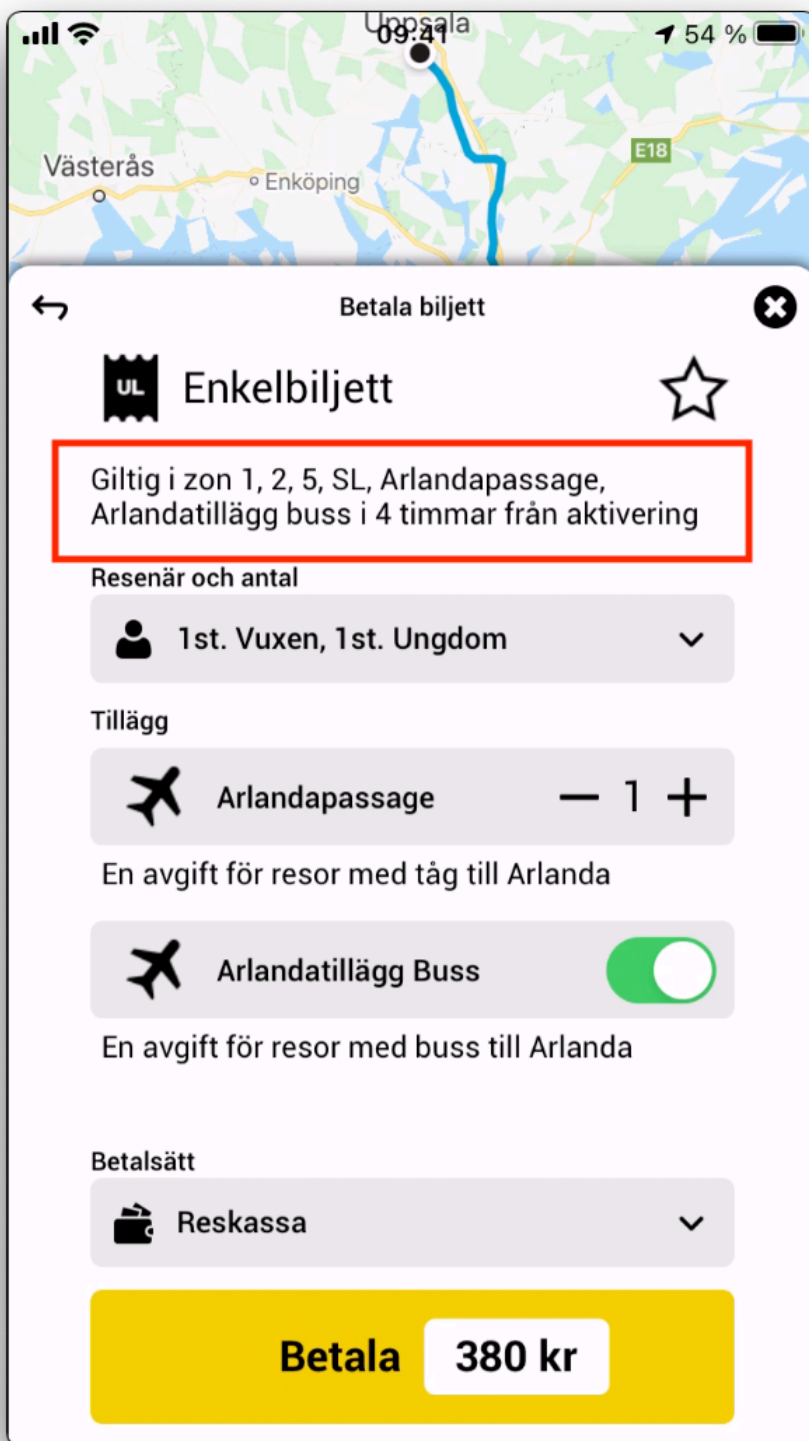
Aria-live anger hur viktiga förändringar i gränssnittet är. Det avgör exempelvis om ett hjälpmedel ska avbryta användaren eller ej.

Använd följande värden:

- Off: Använd för något som uppdateras ofta, exempelvis aktieinformation.
- Polite: Använd om en ändring bara ska meddelas om användaren inte har något viktigt för sig.
- Assertive: Använd när det händer något viktigt som användaren bör uppmärksammas på så snart som möjligt.

Kommentar

Sökförslag och den återstående biljettiden läses upp för en skärmläseanvändare, vilket är jättebra. Det vi saknar är att texten i en köp-process av en biljett också läses upp i samband med att en resenär gör ändringar.



Skärmdumpen är lånad från iOS.

Kopplade riktlinjer

[WCAG 2.0 - 4.1.2 \(A\)](#)

[WCAG 2.1 - 4.1.2 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.4.1.2](#)

Ramar

FR20: Varje rams syfte är beskrivet i attributet title

× Underkänd

Bakgrund

Varje rams syfte måste beskrivas i attributet title. Denna information presenteras av hjälpmedel för gravt synskadade och hjälper dem att förstå om de ska in i ramen och läsa den webbsidan eller fortsätta på den befintliga webbsidan utanför ramen.

Kommentar

Kartfunktionen, som är en tredjepartsfunktion, behöver förklaras med en titel.

Rekommendationer

- Märk upp kartfunktionen med en beskrivande titel genom:
android:contentDescription

Kopplade riktlinjer

[WCAG 2.0 - 4.1.2 \(A\)](#)

[WCAG 2.1 - 4.1.2 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.4.1.2](#)

Sökfunktioner

SF130: Det finns mer än ett sätt att hitta information

✓ Inga fel funna

Bakgrund

För att möta kraven i WCAG 2.1 på nivå AA ska det finnas flera olika sätt att hitta en specifik sida i gränssnittet så länge det inte handlar om steg i en e-tjänst eller process. Om du inte erbjuder en sökfunktion behöver du erbjuda något annat alternativt sätt att hitta en sida på webbplatsen, det kan vara en webbkarta eller en A-Ö-funktion.

Kommentar

Det erbjuds fler alternativ till samma information, punkten är godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 2.4.5 \(AA\)](#)

[WCAG 2.1 - 2.4.5 \(AA\)](#)

[Vägledning för webbutveckling - R32 \(1\)](#)

[EN 301 549 - 9.2.4.5](#)

Dokument och andra medieformat

DM10: Bakgrundsljud kan enkelt stängas av manuellt eller avslutas automatiskt inom 3 sekunder

– Inte aktuellt

Bakgrund

Precis som att rörelser i gränssnittet påverkar besökarens förmåga att ta till sig informationen påverkar även bakgrundsljud koncentrationsförmågan. Därför ska bakgrundsljud som startar automatiskt stanna efter maximalt 3 sekunder, alternativt måste det finnas ett tydligt och lätt sätt att stoppa ljudet.

Kommentar

Vi hittar inget ljud i appen och punkten blir ej akutell.

Kopplade riktlinjer

[WCAG 2.0 - 1.4.2 \(A\)](#)

[WCAG 2.1 - 1.4.2 \(A\)](#)

[Vägledning för webbutveckling - R125 \(1\)](#)

[EN 301 549 - 9.1.4.2](#)

DM20: Det finns en lämplig textbeskrivning av material i komplexa format i anslutning till materialet

– Inte aktuellt

Bakgrund

Komplexa format är ofta bra för användaren. Det ger möjlighet att förklara komplicerade samband på ett mer begripligt sätt. Det är dock inte alltid som användaren kan eller vill ta del av komplexa format. Det kan bero på att användarens utrustning inte stödjer tekniken eller att uppkopplingen är dålig. Det måste därför alltid finnas en tydlig beskrivning av innehållet i dessa komplexa format. När man länkar till exempelvis en film måste man därför beskriva filmens innehåll i text. Hur omfattande beskrivningen är beror helt på filmens relevans. En beskrivning kan också innehålla länkar till andra delar på webbplatsen där man kan läsa motsvarande information som den som ges i filmen.

Ljud

När information presenteras med hjälp av ett ljudklipp måste motsvarande information också finnas i textform på webbplatsen. Det måste också finnas en tydlig koppling mellan ljudklippet och den alternativa texten.

Vid livesändningar ska det åtminstone finnas en tydlig beskrivning av vad det är som sänds, och om möjligt bör också en textåtergivning ges. Det kan exempelvis ske genom att protokollförarens anteckningar vid ett sammanträde samsänds live med ljudet.

Film

För förinspelad film ska det finnas en textversion av innehållet tydligt länkad i anslutning till filmen. En textversion behöver inte vara en ordagrann återgivning av innehållet men ska ge likvärdig information.

Pdf

Även pdf är ett komplext medieformat som ibland skapar svårigheter för användarna. Därför är det även viktigt att ha tydliga länktexter till pdf-filer och att överväga om informationen inte snarare borde ligga i html, eller om det åtminstone ska finnas en sammanfattning av innehållet i html. Tänk också på att själva pdf-filen måste vara tillgänglig.

Kommentar

Punkten är ej aktuell för appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.1.1 \(A\)](#)

[WCAG 2.0 - 1.2.1 \(A\)](#)

[WCAG 2.1 - 1.1.1 \(A\)](#)

[WCAG 2.1 - 1.2.1 \(A\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[Vägledning för webbutveckling - R116 \(1\)](#)

[EN 301 549 - 9.1.1.1](#)

[EN 301 549 - 9.1.2.1](#)

DM30: Ljudinformation i förinspelad video och i ljudklipp återges genom textning

– Inte aktuellt

Bakgrund

Om du har ljud som del av en video eller som en ljudfil så ska innehållet finnas textat. För video räcker det med en textremsa som är synkroniserad med ljudet. Har du ett ljudklipp kan det vara svårt att lägga in en textremsa, då kan du istället erbjuda användaren att läsa en textutskrift av ljudinformationen.

Tänk på att textningen ska återge ljudinformation, det betyder tal men kan även vara annan information. Om det exempelvis ringer på dörren i filmen och skådespelaren därför går till dörren och öppnar ska textningen även återge att det ringer på dörren.

Kommentar

Punkten är ej aktuell för appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.2.2 \(A\)](#)

[WCAG 2.1 - 1.2.2 \(A\)](#)

[Vägledning för webbutveckling - R117 \(1\)](#)

[EN 301 549 - 9.1.2.2](#)

DM40: Ljudinformation i livesändningar återges genom textning

– Inte aktuellt

Bakgrund

Även om du har ljudinformation i livesändningar ska informationen textas. Vi rekommenderar att du gör en rimlighetsbedömning om det är rimligt att erbjuda livetextning i det aktuella fallet, men för att möta kraven i wcag 2.0 på nivå AA krävs det att livesändningar textas.

Kommentar

Punkten är ej aktuell för appen.

Kopplade riktlinjer

[WCAG 2.0 - 1.2.4 \(A\)](#)

[WCAG 2.1 - 1.2.4 \(A\)](#)

[WCAG 2.0 - 1.2.9 \(AAA\)](#)

[WCAG 2.1 - 1.2.9 \(AAA\)](#)

[Vägledning för webbutveckling - R1 \(1\)](#)

[EN 301 549 - 9.1.2.4](#)

DM100: Visuell information i video förklaras med ljud

– Inte aktuellt

Bakgrund

Video ska syntolkas så att en användare som inte ser videon har möjlighet att förstå vad som händer. Det här kan i många fall göras genom att en vanlig speakerröst förklarar vad som sker, men i mer komplexa filmer kan det behövas ett separat ljudspår för ljus syntolkningen.

Även i det här fallet bör du göra en rimlighetsbedömning, men även här är det så att det ska finnas en syntolkning om ni ska möta wcag 2.1 nivå AA.

Kommentar

Ingen video hittas i appen och punkten kan ej bedömas.

Kopplade riktlinjer

[WCAG 2.0 - 1.2.3 \(A\)](#)

[WCAG 2.0 - 1.2.5 \(AA\)](#)

[WCAG 2.1 - 1.2.3 \(A\)](#)

WCAG 2.1 - 1.2.5 (AA)

Vägledning för webbutveckling - R118 (1)

EN 301 549 - 9.1.2.3

EN 301 549 - 9.1.2.5

DM200: Det är tydligt för användarna var de kan hitta alternativ till information som ges på annat sätt än med text

✕ Underkänd

Bakgrund

En film, bild eller ljudfil kan ofta återge komplex information på ett enklare sätt än vad en text kan. Du bör därför sträva efter att använda grafik och ljud för att ge information när det är lämpligt.

Så fort du använder grafik, film eller ljud för att ge information behöver du också ge ett alternativ till de användare som inte kan se eller höra informationen.

Exempelvis kan en film erbjudas i textversion och ett diagram visas som tabell med samma data. För vanliga bilder räcker det med en alt-text (attributet alt). Då ligger beskrivningen direkt på bilden, men så fort du jobbar med mer komplexa alternativ än vanliga alt-texter behöver du tydligt koppla samman de olika alternativen.

Det är ett vanligt problem att alternativa versioner är svåra för användaren att hitta. Grundregeln är att alternativ ska länkas eller presenteras i direkt anslutning till det som de är alternativ till. Till exempel kan flikar bredvid varandra låta användaren växla mellan ett diagram och en tabell. Ett text- eller teckenspråksalternativ till en film bör ligga direkt under filmen. Det är däremot inte tillräckligt att bara lägga in en länk i högerspalten eller förvänta sig att användare hittar till sidan "Tillgänglighet" i menyn.

Kommentar

Se bedömningen från punkt IM70 gällande otillgängliga kartfunktioner.

Kopplade riktlinjer

WCAG 2.0 - 1.1.1 (A)

WCAG 2.1 - 1.1.1 (A)

EN 301 549 - 9.1.1.1

Hjälp och övergång till manuell service

MS40: Om en förklaring finns för hur en användare med hjälpmedel ska hantera en viss funktion bör förklaringen och funktionen vara sammankopplade

– Inte aktuellt

Bakgrund

Ibland behövs en förklaring för gravt synskadade användare om hur en funktion ska användas. Lägg in en sådan förklaring med något av WAI-ARIA-attributen `aria-describedby` eller `aria-label`:

- Använd `aria-describedby` om förklaringen ligger synlig för alla användare. Exempelvis om förklaringen ligger som brödtext under funktionen. Om detta är en förklaring till ett formulärsobjekt bör den i stället ligga i label-elementet. Notera att stödet för `aria-describedby` är bristfälligt idag och att du inte enbart bör förlita dig på det.
- Använd `aria-label` för att lägga in en förklaring som är dold för alla användare utom skärmläsaranvändare.

Kommentar

Vi finner ingen sådan funktion.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.1 \(A\)](#)

[WCAG 2.1 - 1.3.1 \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

Begriplighet

US10: Begripligheten är inte beroende av användarens förmåga att uppfatta olika färger

✓ Inga fel funna

Bakgrund

Många gånger är färg ett tydligt och funktionellt sätt att förmedla information på. Röd färg indikerar varningstexter och felmeddelanden, blå färg är starkt förknippat med länkar. Men även om det är bra att använda olika färger för att förmedla information ska man inte helt förlita sig på färg. Omkring 350 000 svenskar räknas som färgblinda. Det betyder inte att alla ser i gråskala, i den här gruppen ingår också exempelvis alla som enbart är röd-grön färgblinda, men det är en betydande grupp och man behöver inte räknas som färgblind för att ha problem att uppfatta olika färger i olika situationer. Exempelvis är det betydligt svårare än normalt att uppfatta färgändringar i exempelvis löpande text om man sitter med en mobil eller surfplatta utomhus en solig dag.

För att inte användarens förmåga att uppfatta färger ska bli avgörande krävs det att man alltid kompletterar med minst ett annat sätt för att ge informationen.

Länkar kan exempelvis strykas under, knappar i olika färger kan förses med text och så vidare.

Det ska inte heller krävas att användarens dator är "rätt" inställd. I många webbläsare finns en inställning för att "alltid" stryka under länkar vilket medför att om webbplatsen inte har angett att länkar ska vara understrukna eller att de inte ska vara understrukna blir länkarna understrukna. För en person som lånar en publik dator, exempelvis på biblioteket, är det inte alltid möjligt att påverka inställningarna, och man kan då inte vara säker på att "rätt" inställning är gjord. Därför ska man tydligt ange i stilmallarna att länkar ska vara understrukna om det är så man vill märka ut länkar som komplement till exempelvis en specifik färg.

Kommentar

Vi hittar inget innehåll som endast är urskiljbart med färg. Länkar har tillskillnad från iOS-appen en understrkning.

Kopplade riktlinjer

[WCAG 2.0 - 1.4.1 \(A\)](#)

[WCAG 2.1 - 1.4.1 \(A\)](#)

[EN 301 549 - 9.1.4.1](#)

US20: Begripligheten av instruktioner och hänvisningar är inte beroende av användarens förmåga att se och höra

Bakgrund

För att alla besökare ska förstå instruktioner och hänvisningar är det viktigt att dessa inte enbart förlitar sig på besökarens förmåga att se eller höra. Att exempelvis hänvisa med färg kan vara ett bra sätt att förtydliga för användaren, men hänvisningen måste kompletteras med annan information. Istället för att skriva "klicka på den gröna knappen" kan instruktionen lyda "klicka på den gröna knappen med texten 'Acceptera' nedan". Då finns det förutom färgen även en text och en position som hjälper användaren att hitta knappen.

Hänvisa inte heller till objekt som ligger "till höger". Dels går hänvisningen inte att förstå för användare som inte ser gränssnittet. Dessutom hamnar objekt ibland på andra ställen beroende på skärmbredd i responsiva gränssnitt.

De flesta instruktionsfilmer förutsätter att användarna både kan se och höra. Sådana filmer måste därför alltid kompletteras med en textversion.

Kommentar

Vi dedömer inte att instruktioner eller hänvisningar är beroende av specifika kännetecken, punkten blir godkänd.

Kopplade riktlinjer

[WCAG 2.0 - 1.3.3 \(A\)](#)

[WCAG 2.1 - 1.3.3 \(A\)](#)

[Vägledning för webbutveckling - R92 \(4\)](#)

[Vägledning för webbutveckling - R124 \(1\)](#)

[EN 301 549 - 9.1.3.3](#)

Webbtillgänglighetsdirektivet (WAD)

WAD10: Tillgänglighetsredogörelse

× Underkänd

Bakgrund

Enligt Webbtillgänglighetsdirektivet så ska alla webbplatser som omfattas av lagen ha en tillgänglighetsredogörelse publicerad. Tillgänglighetsredogörelse ska vara lätt att hitta på startsidan eller helst åtkomlig från alla sidor, exempelvis i en sidfot. Länken till denna sida bör döpas till "Tillgänglighetsredogörelse".

Redogörelsen ska vara publicerad i ett tillgängligt format och i första hand rikta sig till och ge stöd till slutanvändarna av webbplatsen men även till tillsynsmyndigheten.

Redogörelsen ska innehålla:

- Myndighetens namn.
- Den nationella lagens namn.
- Webbplatsens namn.
- Orsak och aktuell status (Uppfyller kraven helt, Uppfyller kraven delvis eller Uppfyller inte kraven).
- Alternativ till det som inte är tillgängligt.
- Beskrivning av och länk till funktion för återkoppling.
- Länk till klagomålsfunktion hos tillsynsmyndighet.
- Datum för publicering och uppdateringar av redogörelsen.
- Vilken metod har använts för att ta fram redogörelsen.

Rekommendationer (inte krav):

- Länk till granskningsrapport.
- Beskrivning av hur organisationen jobbar med tillgänglighet.
- Formellt godkännande av redogörelsen från ansvarig person i organisationen.
- Datum för när webbplatsen publicerades.
- Datum för senaste väsentliga uppdatering eller redesign.
- Specifik kommunikationstjänst för personer med funktionsnedsättningar som exempelvis Teletal.

Kommentar

Det finns ett avsnitt i tillgänglighetsredogörelsen om appen. Dock täcker den inte alla krav som skall finnas i en tillgänglighetsredogörelse. Bland annat saknas information om vilka krav som appen ej uppfyller.

Rekommendationer

- Komplettera eller ta fram en egen tillgänglighetsredogörelse för appen.

WAD20: Återkopplingsfunktion

✓ Inga fel funna

Bakgrund

Användare ska ges möjlighet att påtala brister kring tillgänglighet genom direkt återkoppling till webbplatsägaren och ansvarig.

- Beskriv funktionen för återkoppling.
- Kontaktuppgifter till avdelning eller person som är ansvarig för webbplatsens tillgänglighet och hantering av återkoppling.

Kommentar

Uppgifter finns i tillgänglighetsredogörelsen.

WAD30: Klagomål till tillsynsmyndighet

✓ Inga fel funna

Bakgrund

I de fall besökaren har återkopplat till webbplatsägare om bristande tillgänglighet men inte fått respons ska det finnas en möjlighet att registrera sitt klagomål till tillsynsmyndigheten.

Det ska beskrivas hur klagomål hanteras i tillgänglighetsredogörelsen samt att det ska länkas till klagomålsfunktion hos tillsynsmyndigheten.

Kommentar

Information finns i tillgänglighetsredogörelsen.

WAD40: Sökfunktion

✓ Inga fel funna

Bakgrund

Många användare ser sökfunktionen som det enklaste sättet att hitta något på en webbplats. För vissa kanske det rent av är det enda sättet. Exempelvis föredrar användare som tycker att det är svårt att stegvis klicka sig fram i en menystruktur att använda sökfunktionen. Sökfunktionen är därför en av de viktigaste funktionerna på en webbplats.

En liten webbplats med ett begränsat innehåll kan klara sig utan en sökfunktion, men det finns ofta ingen större anledning att avstå från att ha en.

Denna punkt sammanfattar sökfunktionen utifrån de bedömda punkterna ovan. Som exempelvis kontraster, korrekt formulärskod och fungerande wai-aria.

Kommentar

Sökfunktion har uppläsning av antalet sökförslag och fungerar väl i allmänhet.

WAD50: Dokument

– Inte aktuellt

Bakgrund

Tillgängligheten i olika dokument, som exempelvis pdf-dokument, är viktig och får inte glömmas bort. I denna punkt granskar vi minst ett dokument per service/huvudingång samt några slumpmässigt utvalda.

Bilden nedan visar ett exempel på hur en automatisk tillgänglighetskontroll av ett pdf-dokument kan se ut i programmet Adobe Acrobat Pro.

Tillgänglighetskontroll

▼

Dokument (6 problem)

✓

Behörighetsflagga för tillgänglighet - Godkänt

✓

PDF-fil med bara bilder - Godkänt

✗

Taggad PDF - Underkänt

?

Logisk läsordning - Kräver manuell kontroll

✗

Primärt språk - Underkänt

✗

Titel - Underkänt

✗

Bokmärken - Underkänt

?

Färgkontrast - Kräver manuell kontroll

> Sidinnehåll (2 problem)

> Formulär

▼ Alternativ text (5 problem)

✗

Alternativ text för figurer - Underkänt

✗

Inkapslad alternativ text - Underkänt

✗

Kopplat till innehåll - Underkänt

✗

Döljer anteckning - Underkänt

>

✗

Alternativ text för andra element - Underkänt

> Tabeller (4 problem)

> Listor (2 problem)

> Rubriker (1 problem)

Ni har mycket att vinna på genom att ställa krav på leverantörerna som levererar pdf-filer till er att filerna också ska vara tillgänglighetssäkrade. I interna system behöver ni också se till att tillgänglighetssäkra era interna mallar som skapar pdf-filer.

Kommentar

Vi finner inga dokument i appen.