



Doro Response - Relatives App - Audit legal requirements - iOS (2025)



Facts about the audit

Client: Doro AB

Fredrik Bengtsson,

fredrik.bengtsson@doro.com

Audit performed by:

Christer Janzon

Our reference:

Lennart Engelhardt,

lennart.engelhardt@funka.com

App: Response by Doro, App Store

Not audited content: Third-party solutions

Time of audit: 2025.03.13 - 2025.04.11

Equipment and assistive technologies used:

- iPhone Xr iOS 18.3.1
- External Bluetooth keyboard
- VoiceOver screen reader

Background

Funka has performed an audit in order to monitor the status of compliance

with the Web Accessibility Directive.

Each success criteria is scored as follows:

- **No errors found.** The interface adequately meets the requirement. This may mean that the solution is very good, or acceptable. There may be room for improvement.
- **No errors found but can be improved.** Either the interface has a few problems in need of fixing (not a consistent error, more a result of carelessness or misunderstandings), or the interface has problems in parts that are beyond the scope of the audit, or problems that for one reason or another needs to be discussed with the client.
- **Fail.** The interface has consistent, clear accessibility issues that need to be rectified.
- **Not applicable.** The interface does not contain the kind of solution the requirement applies to, or the requirement has not been scored for one reason or another.

Funka's methodology is developed in close collaboration with the disability movement and everything we recommend has been tested in a real-world setting. Our operation is based on the international guidelines for accessibility, Web Content Accessibility Guidelines, WCAG 2.2. WCAG 2.2 was launched on October, 2023.

Funka's depth of experience in accessibility and user testing with users with varying abilities and needs, with and without assistive technologies, show that neither the EN or WCAG standards are enough to ensure the accessibility for all users. Because of this, Funka has developed our own requirement to supplement these standards.

Funka has, being named a Lead Translation Organisation, been tasked by the W3C carried out the authorized translation of WCAG 2.0 to Swedish, and has been tasked with the translation of WCAG 2.1 as well.

Funka's specialists are since 2008 involved in the efforts to create the EN301549 standard on behalf of the European Commission. We are appointed experts in the ETSI Special Task Force tasked with harmonizing the EN standard with WCAG 2.1 and ensure that the requirements are applicable on documents and

applications. We are elected technical experts in the Swedish Standards Institute's committee representing Sweden in the standardization efforts. We lead the group of experts that provide support to the member states of the EU and the commission in the transition period for the Web Accessibility Directive and the development of a supervision methodology and accessibility statement.

Audit summary

Many things work very well in the app and we believe that you have given some thought to accessibility during development. However, there are still accessibility issues that need to be addressed, as some user groups may face challenges using the app. Assessments will be a fail even if there is only one issue, though we have seen good examples of accessibility in other parts of the app.

While some problems may be extensive and require significant effort to resolve, there are also smaller issues that present opportunities for improvement.

A major challenge for screen reader users is that some clickable elements and form components are missing names, or have names that are not clear about their purpose. This is true for both iOS and Android.

This becomes problematic already at the login because it's not clear that "caret down" refers to a component that opens a wheel picker for selecting the country code. Additionally, the name itself does not convey meaningful information. This user group will also have major problems with the logged in interface because they cannot control the navigation which slides out from the left. As a result, they are unable to move focus to that area. During our test, we had to stop and restart the screen reader because we got stuck here.

Keyboard users will also experience problems because it is not always possible to control the interface as desired.

Users with visual impairments may experience problems because the contrast is too low for some elements, but also because the content does not adapt to how the user prefers to use the built-in zoom.

The app should also be usable in both portrait and landscape views, but it is currently locked to portrait view only.

One more thing that was cumbersome on iOS was that the info in a SMS is not filled in automatically as it does on Android. If this could be solved here as well, it would make it much easier for certain user groups.

Funka International AB, Stockholm 2025.04.11

Comments

General

GE80: Help functions are consistently placed

✓ No errors found

Background

If you offer help functions such, as contact details, chat systems or self-help guides, these must be consistently placed in the interface. This way, users can always find help in the same place.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

[WCAG 2.2 - 3.2.6 Consistent Help \(A\)](#)

GE90: Pages and functions are described consistently

– Not applicable

Background

The website should be consistent in how it describes different functions, links and areas. For example: if one page says 'e-services', do not call the same services 'self-service' on another page.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 3.2.4 Consistent Identification \(AA\)](#)

[EN 301 549 - 9.3.2.4](#)

Html & css

HC30: The HTML code does not contain critical errors

– Not applicable

Background

There are several different standards in HTML. Each standard has its own rules for which elements and attributes may be used. Some examples of standards are HTML4, HTML5, XHTML 1.0 and XHTML 1.1.

Funka always recommends HTML5 as standard because it has increased accessibility support compared to older standards. Something to keep in mind is that HTML5 is a living standard that is supplemented with elements and attributes on an ongoing basis. For this reason, you should make sure that the new elements and attributes are supported by both browsers and assistive devices before using them.

For browsers, assistive devices, and other tools to interpret the code correctly, they need to know which standard the current page is encoded in. Specify this at the top of the code on all html pages of the website with a !DOCTYPE declaration. !DOCTYPE is declared as follows in html5:

```
<!DOCTYPE html>
```

Follow the chosen standard

However, it is not enough to simply declare a certain standard, it is important to actually follow it. The standard has been created so that developers of, for example, browsers, code libraries, websites, and aids have a common source to look at to ensure that the content is presented optimally.

Robust code

The W3C mentions in WCAG four important specifications that the code must follow in order to be considered robust:

- Complete start- and end tags.
- Do not use the same id value on two different elements on the same page.
- Do not specify the same attribute twice in the same element (eg two alt attributes in the same img element).

- Nest the elements according to the standard.

If there are deviations from these specifications, the interface is not considered robust enough to meet the requirements of WCAG. Other types of deviations from the specifications are not considered to result in as major accessibility issues, but Funka recommends that there should be no deviations from the specifications at all.

There are several different tools to validate the code against the selected standard. W3C has a HTML validator which can be accessed at the address:

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

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 4.1.1 Parsing \(Obsolete and removed\) \(A\)](#)

[EN 301 549 - 9.4.1.1](#)

HC80: The reading order of the content is logical

× Fail

Background

When you control layout and presentation using CSS, you can position different areas visually regardless of their exact location in the HTML code.

However, assistive technology read the website's HTML code from top to bottom. This means that assistive technology presents the content in the same order as the code, not in the visual order determined by the CSS code. For this reason the reading order of the content must be logical and reflect the visual order.

Funka recommends that the user should be able to use the content even when the CSS files are blocked. This enables the user to control how the information is presented.

The fact that it is possible to use does not mean that it is an optimal user experience. For example, it may mean that the menu is presented twice or shows several levels before the user selects something. Of course, such problems must be minimised, but sometimes they may be unavoidable, for example to achieve good functionality when the interface is built responsively.

Apps

The reading order with a screen reader in an app may differ based on the operating system.

iOS

The reading order with VoiceOver is determined on the visual interface. The order is left to right, up to down. Developers can customise the order to match the reading order.

Android

With TalkBack, the reading order follows the order of the code in the same way as in a browser. Again, the order may not match the reading order, so developers can set their own order.

Comment

We find two views in iOS that did not have a logic reading order, see image below. In the first example, "Battery" and its value, as well as "Location" and its value, are not grouped together correctly.

The screen reader currently focuses on icons, which is unnecessary in this type of app since the icons only serve as visual complements to the text. This also slows down navigation for screen reader users because they need more swipe gestures. At moment, the reading order is incorrect: "battery-full" > "map-marker-alt" > "93%" > "Stockholm" > "Battery" > "Location". Not only is the order wrong, but the information is misleading because the icon is announced as "battery full".

The solution is to prevent the screen reader from focusing on decorative icons and also to group the information so that it reads correctly: "Battery: 93%" → "Location: Stockholm".

13:39

98



Overview



Funka Testsson

+46767619313

Doro 8200

updated 5 days ago



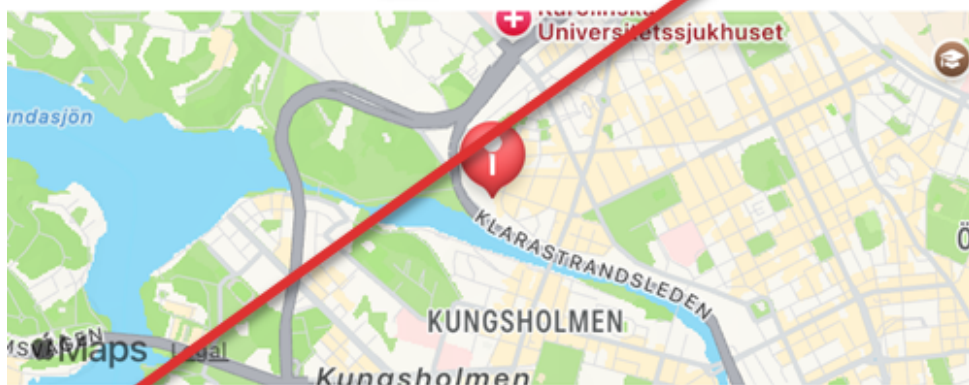
93%

Battery



Stockholm

Location



Location accuracy: Precise



Nybrokajen 7

SE-111 48 Stockholm +46 8555 770 60

contact@funka.com

www.funka.com/en

battery full



This problem was also found in the login with SMS view.



Response by Doro

Please log in to continue



+46

Phone

Log in

Login via email? [Click here](#)

Don't have an account?

[Create new account](#)

We evaluate this criteria as failed.

To do

- Ensure that the information given to a screen reader is presented in a logical order.
- Also make sure that screen readers cannot focus on icons, unless they are the only information carriers.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.2 Meaningful Sequence \(A\)](#)

[EN 301 549 - 9.1.3.2](#)

Flexibility of the interface

RD10: All information can be accessed and used regardless of screen size

! No errors found but can be improved

Background

It is usually impossible to fully predict where, when and how the user will choose to use the interface. For this reason you should never remove the ability to access all information and functionality in the interface even if it is displayed on a small screen. However, feel free to modify the presentation to save space, such as using collapsible areas to hide parts of text.

We usually divide screen widths into three main categories:

- Large screen width - regular computers and laptops with or without an external screen.
- Medium screen width - tablets or computers that use a smaller resolution.
- Small screens - such as mobile phones.

There may be a need for a more detailed division, depending on the interface. It is important that it works well across the spectrum, from small mobiles to large screens, in both portrait and landscape view.

The interface must work down to a width equivalent to 320 CSS pixels and a height equivalent to 256 CSS pixels.

320 x 256 CSS pixels is equivalent to a viewport of 1280 x 1024 CSS pixels at 400% zoom. You can test this yourself by setting the browser window width to

1280 CSS pixels, the height to 1024 CSS pixels, and zooming to 400%.

Row lengths

A recommendation for line lengths for paragraphs is that they should be between 50 and 80 characters long including spaces in all screen sizes.

There are different principles and methods that can be used to build responsive solutions. If line lengths and areas adapt to the window width, that's good, but the most important thing is that the interface is usable from smaller screens to very wide screens, and that the presentation order is logical regardless of screen width. This also includes ensuring that line lengths are sufficient to accommodate long words.

Navigation

As soon as visible navigation collapses into a so-called hamburger menu, the interface generally becomes more difficult to use. A recommendation is therefore to try to maintain navigation where level one is visible in as many screen sizes as possible.

If you simplify so that not all information or services can be accessed in one screen size, you must offer the user the possibility to get to the regular interface.

Comment

We have not had the opportunity to test the app on a device other than the iPhone on which this audit is based. However, we recommend testing the app on other Apple devices with different screen sizes to make sure the app works well there as well.

Therefore, this criterion evaluates that no errors have been found, but that there may be room for improvement.

To do

- Keep in mind that an iPad-adapted version can give the users a better experience.

Links to guidelines

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

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

[WCAG 2.2 - 1.4.8 Visual Presentation \(AAA\)](#)

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

RD30: The interface is viewable and usable without forcing the user to scroll in more than one direction

✓ No errors found

Background

It is impossible to predict how a user will choose to view or use an interface. For this reason the interface must be responsive. It should adapt to the screen width chosen by the user, without losing content or functionality.

If content or functions do not fit on the user's screen, scrolling is only allowed in the vertical or horizontal direction, not in both directions at the same time. The interface must, under this condition, work from at least 320 pixels wide or 256 pixels high.

Sometimes exceptions may be necessary for specific content, such as tables, charts or illustrations.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

WCAG 2.1 - 1.4.10 (AA)

WCAG 2.2 - 1.4.10 Reflow (AA)

EN 301 549 - 9.1.4.10

RD40: The website is fully legible and usable when zoomed 200%

✗ Fail

Background

Some browsers offer the possibility of adapting the text size if the texts are inserted with relative sizes (e.g. em and %). This setting has been overshadowed these days by the more widespread zoom function that magnifies everything, regardless of what dimension units the text is in.

Although websites no longer have to be built using relative dimensions, it is important that text does not become difficult to read and that functions are difficult to use if the user has changed the text size settings in the browser. If the website

is designed with relative sizes, it also means this needs to be done in a way that does not distort the way the pages are displayed in the browser.

Users must also be able to zoom the website up to 200% without texts overlapping each other and becoming difficult to read or functions becoming difficult to use. This does not just apply to large screens with HD resolution but also to smaller screens and smartphones.

Bear in mind that a responsive user interface should adapt when the user zooms. In other words, even when you zoom in on a responsive user interface, you should be able to access all functions and all the information. Naturally, there is a limit where too small a screen with too great a zoom will no longer work, but when the user reaches this point, the user should still be able to continue to zoom even if this can mean having to scroll sideways to read all the text.

You will ensure users zoom in on mobile devices with the following code in the page header:

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

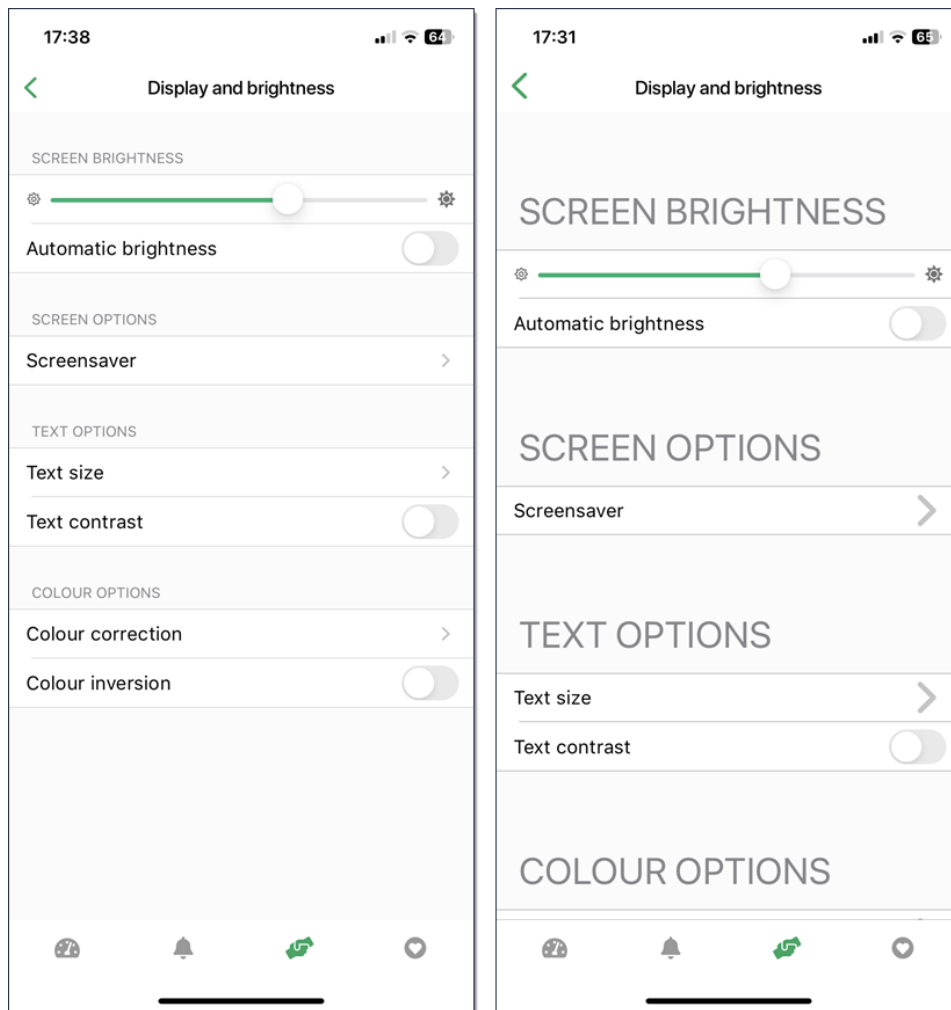
The content attribute can contain many more sub-attributes and associated values to control the presentation on different screens. Using the maximum-scale sub-attribute can impair user scope to zoom even when you have stated user-scalable="yes". We recommend you avoid using maximum-scale. To date, we have not experienced any problems with other sub-attributes.

Although a specific solution for how to manage to zoom and change window size must be customised from case to case, here are a few brief recommendations for normal web interfaces:

- The website should adapt automatically to the window size, but it is important to insert mechanisms that prevent line lengths longer than 70 characters, including spaces or being so short that whole words cannot be displayed.
- If the user changes the text zoom function in the browser, this should not mean that the content cannot be read or functions are not able to be used.
- Zooming via the browser zoom function should work up to at least 200%.
- If the window or text size is reduced, you should still be able to read the text and use the functions.

Comment

It is important for users to be able to change the text size while ensuring the app remains usable. When we increase the text size to the equivalent of 200%, the content is only enlarged in one type of view and that is where the user change the user settings "Display and Brightness" and "Sound and Vibration". There, the headlines are enlarged as they should be, but nothing else in the app is enlarged, except for the notifications that use the system's settings, see image below.



17:30



Personal information



Change profile picture

Tap here to change



Name



**Want to change
your phone
number?**

This action is only
available via Doro
Support

OK

We evaluate this criteria as failed.

To do

- Ensure that the text supports Dynamic Type so that it is affected when zoomed.
- Ensure that the text container is flexible so that it adapts when the text is enlarged.

Links to guidelines

[WCAG 2.0 - 1.4.4 \(AA\)](#)

[WCAG 2.1 - 1.4.4 \(AA\)](#)

[WCAG 2.2 - 1.4.4 Resize Text \(AA\)](#)

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

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

[WCAG 2.2 - 1.4.8 Visual Presentation \(AAA\)](#)

[EN 301 549 - 9.1.4.4](#)

RD60: Users can access all content and use all functions regardless of screen orientation

× Fail

Background

When content is presented to a user, it should not be locked to a portrait or landscape view. Videos and tables should be responsive and viewable in any orientation, unless there are specific reasons. Full-screen videos often require the device to have a landscape orientation, but should be playable in portrait view as well.

In various situations the user may need to use mobile phones or tablets in a specific orientation, for example if the device is mounted to a holder on a wheelchair. The orientation of the interface should then be adapted to the user's conditions. As a result, the experience is not always optimal, but it should work. An example of this is viewing videos in portrait mode.

This requirement does not apply to changes in content or features due to screen size, only to screen orientation.

Comment

The app is locked to portrait mode, which means that this checkpoint fails. To meet WCAG Success Criterion 1.3.4 , Orientation, the app must not restrict its view and operation to only a single display orientation, such as portrait mode.

To do

- Enable to use the app in both portrait and landscape mode.

Links to guidelines

[WCAG 2.1 - 1.3.4 \(AA\)](#)

[WCAG 2.2 - 1.3.4 Orientation \(AA\)](#)

[EN 301 549 - 9.1.3.4](#)

Navigation & links

NL20: The interface can be controlled by keyboard on both desktop and mobile

✕ Fail

Background

There are different ways to control a computer or smartphone. There are different solutions for moving the mouse cursor, clicking, and entering text. By enabling the interface to be controlled by keyboard, mouse and touchscreen, the user can choose the solution they prefer.

For some users, there are no options. Some cannot use a regular keyboard and must navigate with the mouse. Other users cannot use any kind of mouse or touchscreen and must navigate with the keyboard.

A basic principle in WCAG is that it should be possible to control the interface using only a keyboard, whether it is on a large or small screen. This must work if you are to be considered compliant with WCAG.

Exceptions to this requirement can be made if, for example, it concerns a function that could not reasonably be made to work independently of the input device.

Avoid common problems

Avoid adding features that require the user to click or hover over different areas, such as menus that require the user to hover with the mouse.

Avoid lists that refresh the page or move the focus as soon as the value in the list changes. These types of solutions create major problems especially for people who have to navigate with the keyboard, but also for people with reduced precision or who are sitting in busy environments.

Be careful about changing interaction patterns that the user is familiar with. For example, users who navigate with the keyboard are used to using the tab key to

move between links and form items. If the user is suddenly expected to use the arrow keys instead, many users will have problems.

You should also bear in mind that the combination of responsive design and zoom can result in mobile viewing even for desktop users. It is then essential that it works independently of the input device also in mobile view.

Comment

When using an external keyboard to navigate, we encountered a few different serious problems.

When we use an external keyboard to navigate, we had difficulty both in getting focus on some clickable objects but also in not being able to get focus in the menu (left sliding menu).

Our first example concerns the problem of getting focus in the menu (left-sliding menu), which means that it is also not possible to access the clickable options "Settings" and "Help", see image below.

Another problem with this view is that the menu automatically slides out when the keyboard focus is on one of the two buttons that can open it. This should not happen, a menu should open when the user has actively chosen to open it.

17:03



Mia Testsson



Settings

Seniors



Funka Testsson



Help



A
C



A
C



A
C



In the "Responders" view, we were unable to set the focus on the pen icon near the respondent.

16:49



Responders



SENIOR NETWORK



PENDING

This is the Group of Responders for Funka Testsson who will receive alarms.



Mia Testsson (you) ✓

+46767619329



Christer Janzon ✓

+46708231065

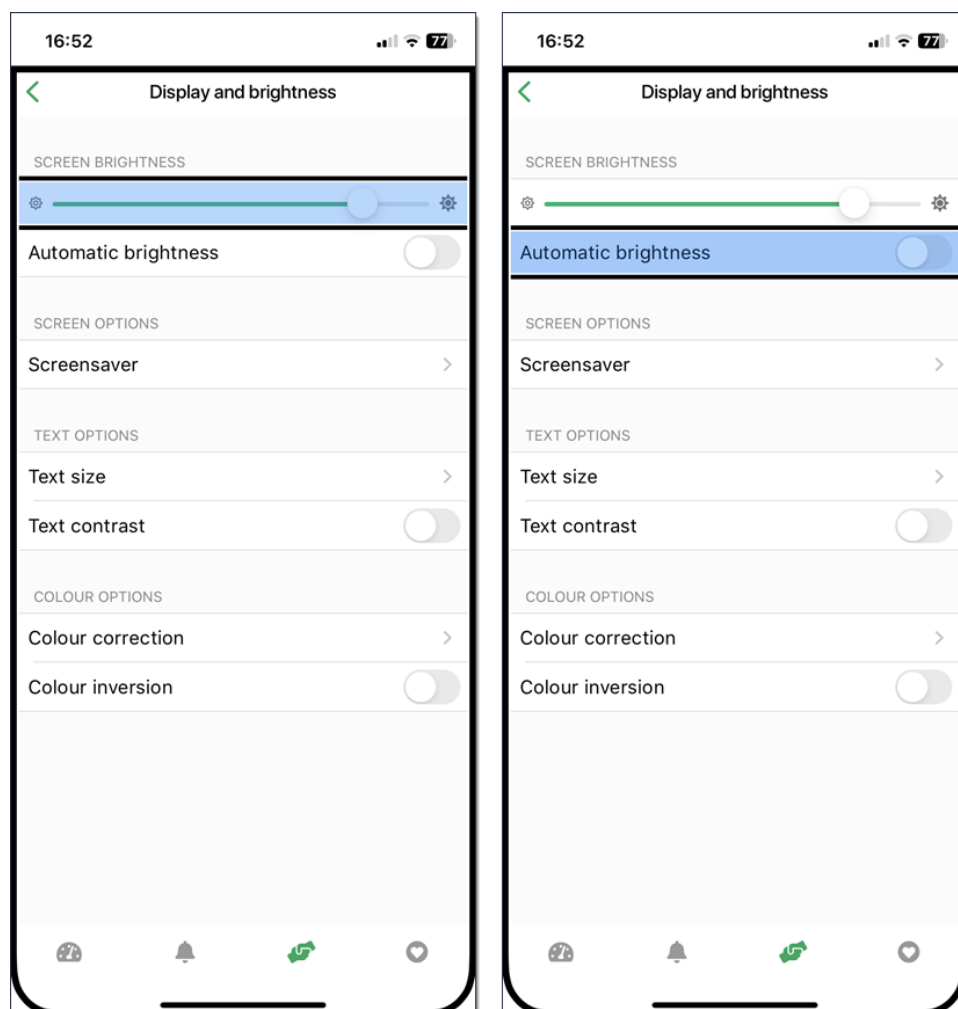


INVITE



Finally, we noticed issues with managing the slider under "Display and Brightness," while the sliders under "Sound and Vibration" functioned as expected.

In both of these views we also found that managing the switch buttons did not work, what happens is that the screen flashes and the focus moves up to the first slider.



We evaluate this criteria as failed.

To do

- Ensure that all interactive elements can be accessed and activated using the keyboard.
- Also, ensure that the menu does not open automatically when a button receives keyboard focus. Make sure the menu can be easily focused.
- Try testing it yourself by connecting an external keyboard to your device and navigating through the app. Tabbing should be smooth, follow a logical order,

and it should be clear where the focus is at all times.

Links to guidelines

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

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

[WCAG 2.2 - 2.1.1 Keyboard \(A\)](#)

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

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

[WCAG 2.2 - 2.1.2 No Keyboard Trap \(A\)](#)

[WCAG 2.2 - 2.1.3 Keyboard \(No Exception\) \(AAA\)](#)

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

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

[WCAG 2.2 - 3.2.1 On Focus \(A\)](#)

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

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

[WCAG 2.2 - 3.2.2 On Input \(A\)](#)

[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: Keyboard navigation follows a logical order

✓ No errors found

Background

A user navigating with a keyboard cannot control the order in the same way as a user navigating with a mouse or touchscreen. The navigation order is instead controlled by the website.

This is often only related to the tab order. The most common way to jump between items with the keyboard is to press the tab key repeatedly. This moves the focus between the different links and form objects on the website.

The tab order is normally the same as the order of the content in the code. Each time the user presses the tab key, they jump from one link or form object to the next.

The attribute `tabindex` can be used to control the tab order. There may be occasional reasons to use this attribute, for example to help prevent an object from gaining focus, but all objects that the user tabs to should be in a logical order

in the structure of the page. In this case, there is no need for a tabindex to control the order.

Tabindex works as follows:

- **Tabindex = 0** adds inactive objects, such as a div, to the tab order where they are naturally located in the page structure. Funka's recommendation is to primarily use HTML elements that are naturally interactive (such as links and buttons). However, in some situations there may be a need to allow the user to tab to other objects, in which case `tabindex="0"` should be used.
- **Tabindex > 0** prioritises the items and changes the navigation order so that it does not follow the structure of the page. The order will be from the lowest number upwards. After the user reaches the item with the highest tabindex entry, the tab order continues from the first interactive item in the code.
- **Tabindex = -1** means that the object cannot be tabbed to, but that it can be given focus through scripts. Used, for example, to script focus on an error message so that it is conveyed to assistive technology users.

In some functions, the user navigates with other keys as well, such as the arrow key to move focus between different radio buttons. It is important that other keyboard navigation has a logical sequence as well.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 2.4.3 Focus Order \(A\)](#)

[EN 301 549 - 9.2.4.3](#)

NL40: Focus is clearly displayed when users navigate with the keyboard

✓ No errors found

Background

When users navigate with the keyboard on a web page or in a function, it must be visually clear where the focus is. When you tab through the menu, the menu option that is currently in focus can be highlighted with a frame or by inverting the colours. In some functions where you can navigate with the arrow keys, the same recommendation applies: The object that is in focus should be clearly delineated visually.

Hidden links and shortcuts should also be highlighted and displayed visually when they receive focus with keyboard navigation.

The focus indicator should be visible. For example, it can be a slightly thicker frame, a thicker underline, an inversion of colours or a background plate that appears.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 2.4.7 Focus Visible \(AA\)](#)

[EN 301 549 - 9.2.4.7](#)

NL50 : When an object receives keyboard focus, it is at least partially visible

✓ No errors found

Background

For sighted users navigating the interface with a keyboard, it is crucial to understand where the focus is. Therefore, it is important that the object with keyboard focus is always visible to users. This applies to all screen sizes.

To meet this requirement, the object with keyboard focus should be at least partially visible. The recommendation is that the entire object is visible. Common areas that may obscure focusable objects are headers, navigation elements, and

modals that do not scroll with the rest of the page but remain in their position, known as "sticky".

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

[WCAG 2.2 - 2.4.11 Focus Not Obscured \(Minimum\) \(AA\)](#)

NL60: There are shortcuts to facilitate keyboard navigation

– Not applicable

Background

For users who rely on keyboard navigation, it can be inconvenient to tab through the entire menu structure on every page. When users have found the page they were looking for, they are interested in the content.

To make it easier for these users, shortcuts should be provided on the pages. The most common shortcut leads directly to the content of the page, bypassing the navigation. But in different contexts it may also be appropriate to have links back to the top of the page, or to different menu groups on the page.

These types of shortcuts are created by inserting links that lead to an element further down the page. Visually, these types of links should be hidden until a user gets keyboard focus on them.

Links to content are created like regular links but with the # character followed by the target's id value. The most common is to link to the content area or the initial heading.

The link can look like this:

```
<a href="#content">
  Skip to main content
</a>
```

The target, which in this example is a heading, can look like this:

```
<main>
  <h1 id="content">
```

```
Content main heading
</h1>
</main>
```

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 2.4.1 Bypass Blocks \(A\)](#)

[EN 301 549 - 9.2.4.1](#)

NL70: Keyboard shortcuts are not only activated with alpha-numerical keys

– Not applicable

Background

Many users control the user interface entirely or partly with keyboard navigation and keyboard shortcuts. For some users with mobility disabilities or severe visual impairments, the keyboard is the primary form of controlling the interface. With voice control, users may use voice commands to then convert these into keyboard commands. For example, a person that controls the interface via the voice can say "tab" to move to the next link. This guides the assistive technology to send the keyboard command "tab" to the interface.

It's important that the keyboard shortcuts and regular shortcuts are not based on numbers, characters, or symbols only, since this may cause users navigating with the voice or keyboard to accidentally activate functions. Examples of this may be that information is deleted, or contacts are being called by mistake by the user.

When using this kind of keyboard shortcut, the following must be taken into consideration:

- There's a mechanism to turn off keyboard shortcuts.
- Keyboard shortcuts can be changed - to be controlled by another key.
- The keyboard shortcut is only possible to use when the controlling object is in focus.

The accesskey attribute is not covered by this requirement since that kind of

keyboard shortcut is activated by pressing one key and one of the Alt, Shift, Control or Option keys simultaneously.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

[WCAG 2.1 - 2.1.4 \(A\)](#)

[WCAG 2.2 - 2.1.4 Character Key Shortcuts \(A\)](#)

[EN 301 549 - 9.2.1.4](#)

NL90: Clicks can be cancelled or the user has an option to undo the click

✓ No errors found

Background

A click consists of two parts, a down event, and an up event. The down event occurs when the user pushes their mouse button, or their finger down on the screen. The up event is when the button is released, or the finger is lifted from the screen.

When a user navigates through an interface by clicking with a mouse cursor or a touchscreen, unintended events sometimes occur - the user simply clicks or presses in the wrong place.

To reduce such problems: avoid functionality that happens when the user clicks down. Instead, let the functionality be activated when the user releases the mouse button.

In some cases this is not possible, such as for a piano app, where the user plays the piano by clicking down on the keys.

If an event must be activated by a down event, there should be a chance for the user to undo the event if possible. An example of this is many on-screen mobile phone keyboards. By resting the finger on a letter, a number of less common letters appear (example: if you rest your finger over the A key, you can type á, â or). If the user lets go of the screen without making a selection, no character is typed.

Sök resa Nästa avgång

Från station/hållplats/adress

Använd min plats

Till station/hållplats/adress

1 2 3 4 5 6 7 8 9 0

ß š ś š š

Q I O P Å

A S D F G H J K L Ö Ä

↑ Z X C V B N M ↵

!# 😊 😊 Svenska . Gå

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

[WCAG 2.2 - 2.5.2 Pointer Cancellation \(A\)](#)

[EN 301 549 - 9.2.5.2](#)

NL100: Content and functions are usable without complicated gestures

✓ No errors found

Background

Users with reduced mobility in their hands and fingers have difficulty controlling the precision that is often required to navigate an interface. There are also input techniques where difficult gestures, such as eye tracking, cannot be performed. For example, it is very difficult or impossible to perform gestures that require more than one touch point with eye tracking, or to achieve a motion path with a switch. For this reason, it is important to create interfaces based on the simplest possible gestures. There should be alternative ways that allow these users to easily control the interface without difficult gestures. Users with a head mouse or eye tracker are not helped by keyboard shortcuts via an on-screen keyboard. Instead, gestures should be complemented by buttons that the user can press.

Examples of one type of interface based on gestures are server-based maps, where the user can zoom in and out by pinching together or apart with two fingers. Another gesture is to use two fingers to move the geographical area displayed on the map horizontally or vertically.

Alternatively, for users with motor disabilities, the zoom function can be controlled by buttons: plus to zoom in and minus to zoom out. Right, left, up and down buttons allow the user to tilt and pan the map to change the geographical area.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

[WCAG 2.1 - 2.5.1 \(A\)](#)

[WCAG 2.2 - 2.5.1 Pointer Gestures \(A\)](#)

[EN 301 549 - 9.2.5.1](#)

NL105: Features can be used without dragging motions

× Fail

Background

Users with reduced mobility may have difficulty controlling certain interfaces where a dragging motion is required.

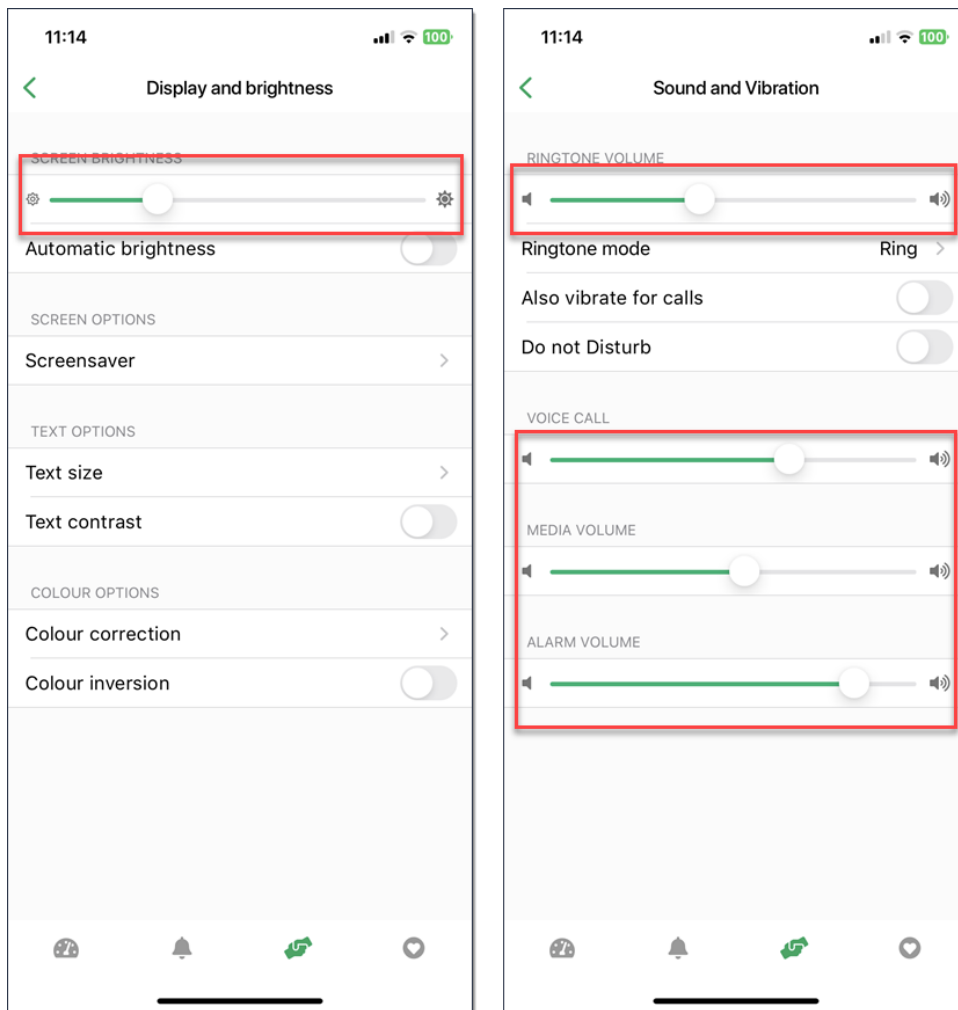
Additionally, dragging motions cannot be performed with certain techniques, such as eye tracking.

If a solution requires the user to perform a dragging motion, there must be an alternative solution, such as buttons.

Comment

There are functions where the user must be able to perform a dragging gesture. In the examples below, the user needs to hold down a finger and drag left or right to adjust the setting.

On Android, this works because it is also possible to tap on the slider to select a value. On iOS, this is not possible.



We evaluate this criteria as failed.

To do

- Modify the slider so it is also possible to tap on the slider to make adjustments.

Links to guidelines

[WCAG 2.2 - 2.5.7 Dragging Movements \(AA\)](#)

NL110: Visible descriptions of clickable objects are described in a similar way through hidden descriptions in the code

✓ No errors found

Background

There are many techniques for describing interactive objects. For common input fields, a visible label is often used in connection with the input field. It is usually enough if the label is inserted correctly, but in some situations, visually hidden descriptions inserted with, for example, the aria-label attribute is also used. There may be many reasons for adding a visually hidden description. For example, a user who doesn't see the interface may be in need of an extended description of the field.

When using hidden descriptions it is important that they describe what is displayed visually. An object with a label containing the text "Buy" or "Read more", or an image illustrating the function of the object, should begin with the same text in the visually hidden label or description. A button with the description "Buy ticket" should, for example, also be initiated with the wording "Buy ticket" in the aria-label or aria-labelledby attributes.

Users with cognitive or severe mobility disabilities may use their voice to control the interface, rather than using the keyboard or pointing device to navigate. When visually visible and hidden labels match, voice control also works well and is predictable.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

[WCAG 2.2 - 2.5.3 Label in Name \(A\)](#)

[EN 301 549 - 9.2.5.3](#)

NL120: Functionality that is activated through motion can be disabled and controlled in a different way

– Not applicable

Background

Mobile devices can for example use accelerometers, gyroscopes and cameras as an input method or to perform a function.

For users with mobility disabilities, it is problematic when functions in an interface are based on motion. This assumes that the user can move their device freely, which is not possible for example for users who have their device mounted on their wheelchair.

For this reason, interfaces should, whenever possible, be controlled by buttons, links and other objects as an alternative to movement.

There are exceptions for functions that rely on movement to operate, such as pedometers and spirit levels.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

[WCAG 2.2 - 2.5.4 Motion Actuation \(A\)](#)

[EN 301 549 - 9.2.5.4](#)

NL160: Target size meet the minimum size requirement

✓ No errors found

Background

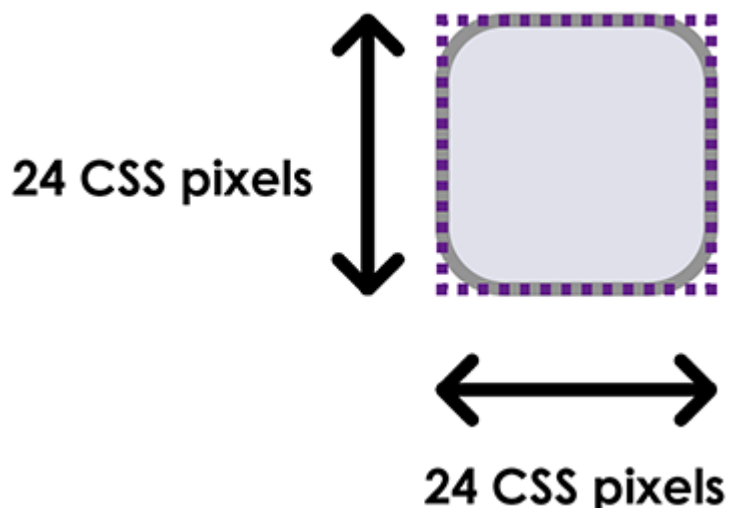
Adequately sized clickable areas are important for all of us to hit the right spot, but for some users, it's especially crucial. This is the case, for example, when using a touchscreen, having impaired precision, or being in distracting environments, such as on a train.

As important as the size of clickable areas is the distance between them. The issue for users is often not small clickable areas but rather that they are placed too closely together. If two clickable objects are placed next to each other, either the distance between them should be sufficiently large, or the clickable areas

should extend relatively far in each direction so that the user doesn't risk hitting the wrong area.

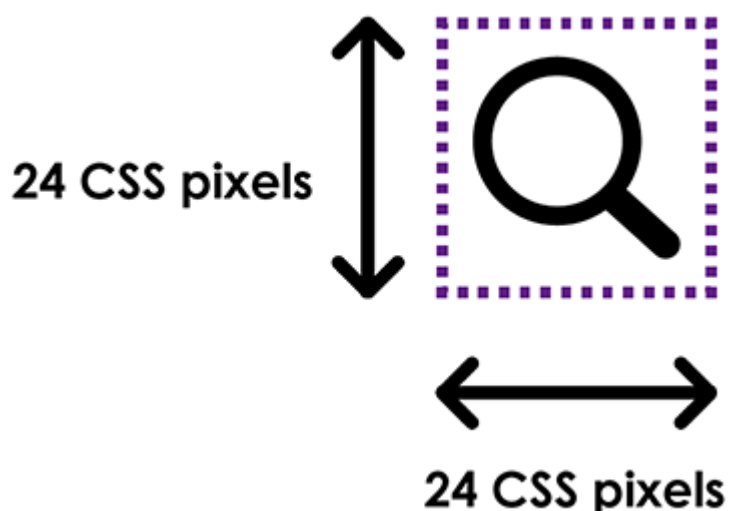
The minimum requirement for the size of clickable areas is at least 24 by 24 CSS pixels. This means that all sides of the clickable area should be at least 24 CSS pixels long. There are some exceptions to this requirement, such as links within body text or if the clickable areas are sufficiently spaced apart.

Example of a clickable area that does not meet the minimum size requirement:



The rounded corners result in a square with a side length of 24 CSS pixels not fitting entirely inside the clickable area. If the distance to other clickable areas is sufficiently large, the example above could still be approved.

Example of a clickable area that meets the minimum size requirement:



A square with a side length of 24 CSS pixels fits entirely inside the clickable area.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

[WCAG 2.2 - 2.5.8 Target Size \(Minimum\) \(A\)](#)

NL170: Link groups and information areas are grouped

✓ No errors found

Background

To make it easier for the user to identify groups of objects, it is important that there is a clear grouping in the code, for example, by grouping menus using bulleted lists. The menu links will then be presented as a coherent group in assistive technologies used by severely visually impaired users. In menus where submenus are expanded, nested lists should be used (lists within lists). The hierarchical structure will then be perceivable by severely visually impaired users.

In HTML5, there is an element called nav that should be used to declare the navigation on the page. The element should enclose both the main menu and submenu, or two different nav elements should be used to declare the main menu and submenu, respectively. The element can also be used for a larger menu in the page footer. This is how it is used:

```
<nav> ... Menu links ... </nav>
```

Landmarks

WAI-ARIA defines different landmarks that are also used to mark up different types of content on a page. These will add more structure for users that cannot see the interface.

If you are building with HTML5 you should use the elements available in HTML5 firstly. You can also insert landmarks but avoid using the element nav and role="navigation" at the same time, this can cause problems for screen readers in certain cases.

How to insert landmarks:

```
<div role="search"> ... Search function ... </div>
```

Use the following landmarks where relevant:

- **role="search"**, which marks the search function
- **role="main" or <main>**, which marks the main content on the page
- **role="complementary"**, used for content that is complementary to the main content on the page. This includes all types of "related information", i.e. the type of content that typically is placed in the right-hand column on the page.
- **role="banner" or <header>**, used for the page header
- **role="contentinfo" or <footer>**, used for identifying information, such as copyright information, navigation links, and privacy statements. This section is commonly called the page footer.

There are several landmarks:

- **role="application"**, used for areas with application-like interactions, for instance, a text editor. Avoid using this landmark unless you have very strong reasons for using it.
- **role="form"**, avoid this as it does not add anything for users at the moment.

[Information about WAI-ARIA \(W3C website\).](#)

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

NL190: The target of links is clearly stated in context

– Not applicable

Background

When link texts are clearly formulated, it will help guide all users to find the right link. A clearly formulated link is also a necessity to enable severely visually impaired users and certain users with cognitive disabilities to find the information they are looking for. Links must therefore provide clear information as to which pages they lead to and what type of content these pages contain.

Severely visually impaired users find it impossible to get a visual overview of the

page. This means that they cannot identify the links located on the page at a glance and what blocks of text they are linked to. Severely visually impaired users have to navigate their way around the page instead of exploring the page with the support of their assistive technology, a screen reader.

A screen reader only presents the information the user is focusing on. If the focus is on a link, only the link will be read, not the text around it. The user cannot visually connect the link to the adjacent text. This means that the link itself must provide all relevant information as to where it leads.

Many tools also include a function that takes the links out of their context and presents them in a separate link list. This speeds up navigation and enables the user to quickly find specific links like "Log in" or "Shopping basket". When a link is taken out of context, it does not help the user if the text next to the link says where it leads, but the target of the link must be clear from the link text itself. Links that say "Read more" or "Click here" are therefore not sufficient.

Linked images must provide all relevant information about the link in the alt text (sometimes called "description" or "tooltip" in different authoring tools).

For email links, it is important that the actual email address is stated in the link text, plus information about the recipient, for instance, the recipient's role and/or name. Stating the email address is important to enabling a user who does not wish to email straight away to copy the address to use at a later time.

Title text

Do not insert any title text (sometimes called "description" or "tooltip") in links. The link text should give the user all relevant information about the link. Some websites used to add a title text to the link that would be shown when the user hovered over the link, but this solution cannot be utilized by users who use a keyboard or touchscreen navigation.

Links are understandable in their context

WCAG on level A requires links to be understandable on their own or through their programmatically determined context. That means that it should be possible to understand the link together with any text in the same HTML element.

That could mean placing a text and the link in the same paragraph element or list element.

The easiest way to meet this requirement is, however, what's easiest for the

users; to write link texts that make it easy to understand the link's purpose regardless of context.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 2.4.4 Link Purpose \(In Context\) \(A\)](#)

[EN 301 549 - 9.2.4.4](#)

NL360: When the same menu or link collection is shown on different pages, the sorting order should be consistent, unless users would find the order illogical

– Not applicable

Background

Do not change the sort order in the listing of links, tools, or menus on different pages. The same menu or link collection should be presented in the same relative order throughout the website unless it would not be perceived as illogical by the user.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[EN 301 549 - 9.3.2.3](#)

Colours & presentation

CP10: The text contrast between foreground and background is adequate

× Fail

Background

To enable visitors to access the content in a plain and clear way, it is important that there are good contrasts (actually light contrast) between the text and background. On a contrast scale that goes from 1:1 (no difference) to 21:1 (black on white), the contrast ratio should be at a minimum of 4.5:1.

For larger text, such as headings, a contrast ratio of at least 3:1 is enough. Though our recommendation is to keep the contrast ratio at least at 4.5:1, for the text to be clear for as many users as possible.

	24 pixels or less, or 18.5 pixels bold or less	Larger than 24 pixels, or larger than 18.5 pixels bold	Graphical objects, clickable surfaces, form fields, diagrams, and similar.
Fail	Approve	Approve	
Approve			

Also, keep in mind that the contrast requirement applies to text that appears in videos, such as nameplates and subtitles.

Tools for measuring contrasts

To measure contrasts, a complex formula that is available on the WCAG website is used:

[The formula for contrasts \(WCAG 2.1 website\), opens in a new window](#)

If you don't want to calculate this yourself, WAI recommends the tool "Colour Contrast Analyser", which can be downloaded free from the TPGi website:

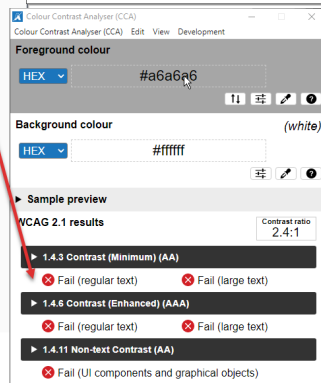
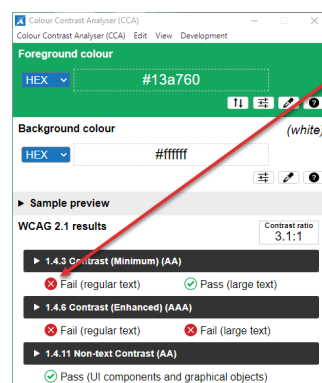
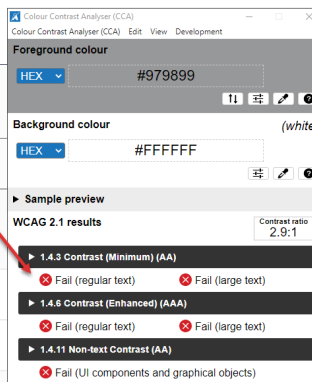
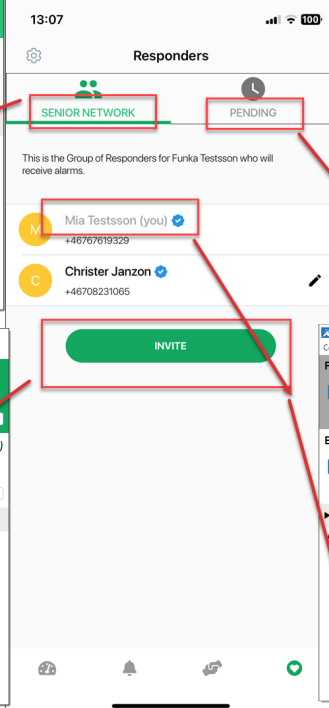
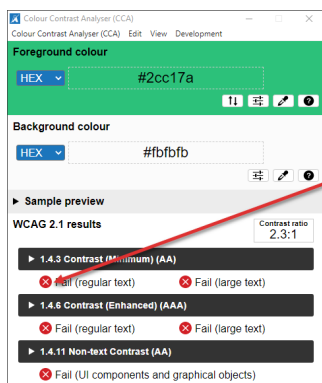
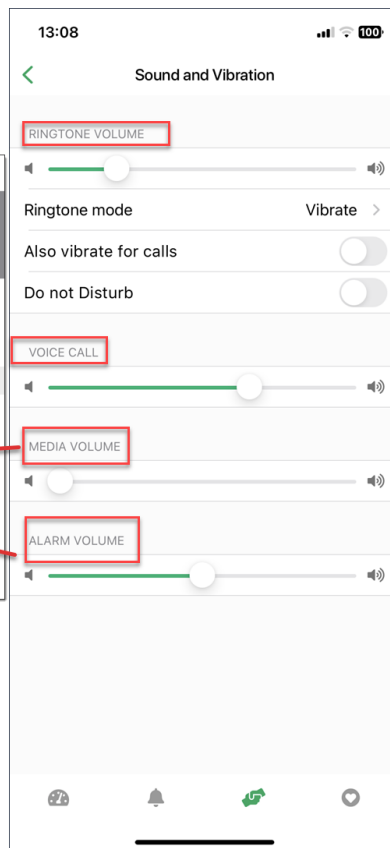
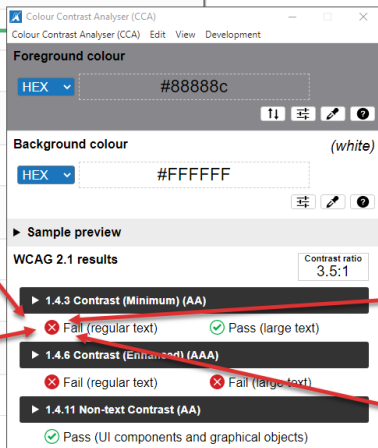
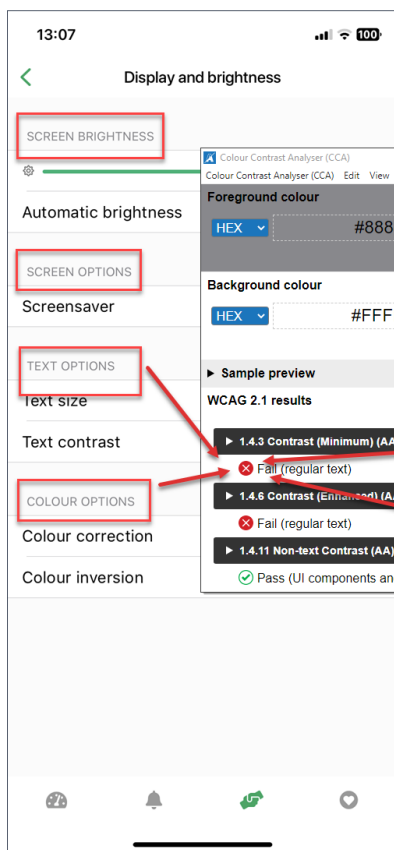
[Colour Contrast Analyser \(TPGi website\), opens in a new window](#)

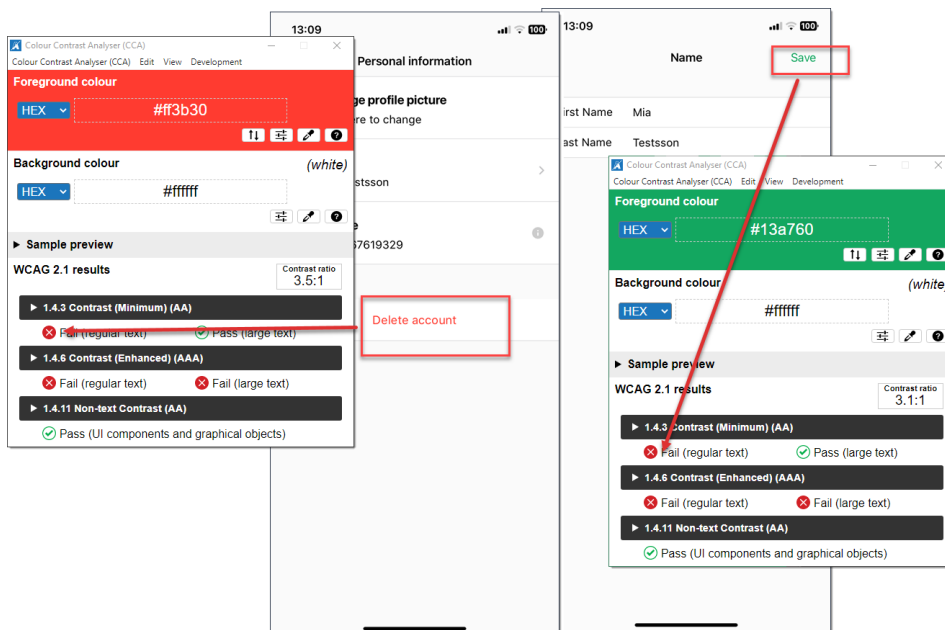
Comment

To comply with WCAG 1.4.3, the contrast ratio between text and background must be at least 4.5:1 for small text and 3.0:1 for large text. Unfortunately, we found that contrast falls below these thresholds little bit here and there. Proper text contrast is particularly important for users with visual impairments.

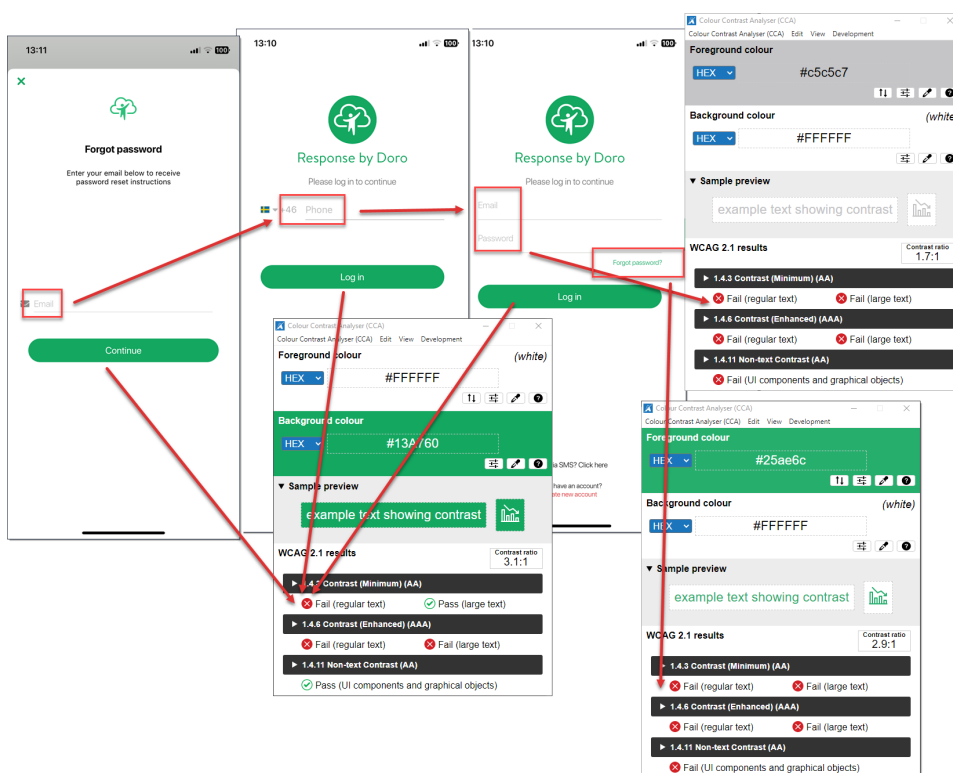
The green colors you use against white text, or vice versa, do not meet WCAG requirements for text contrast, and they also differ from the colors used on the website. This contrast problems exists before and after login. We also found contrast issues with other colors.

The example images below illustrate where the contrast requirements are not met.





In the login function itself, there is also text that has a too weak contrast ratio against the background, see image below.



We evaluate this criteria as failed.

To do

- Ensure that the green color you use is darker so that white text passes the contrast value if used together.
- Ensure that all text in Doro Resonse apps has sufficient contrast against the background color. Our recommendation is a minimum contrast ratio of 4.5:1 for all text.

Links to guidelines

[WCAG 2.0 - 1.4.3 \(AA\)](#)

[WCAG 2.1 - 1.4.3 \(AA\)](#)

[WCAG 2.2 - 1.4.3 Contrast \(Minimum\) \(AA\)](#)

[WCAG 2.2 - 1.4.6 Contrast \(Enhanced\) \(AAA\)](#)

[EN 301 549 - 9.1.4.3](#)

CP20: Graphical objects conveying information, illustrations and videos have adequate contrasts

× Fail

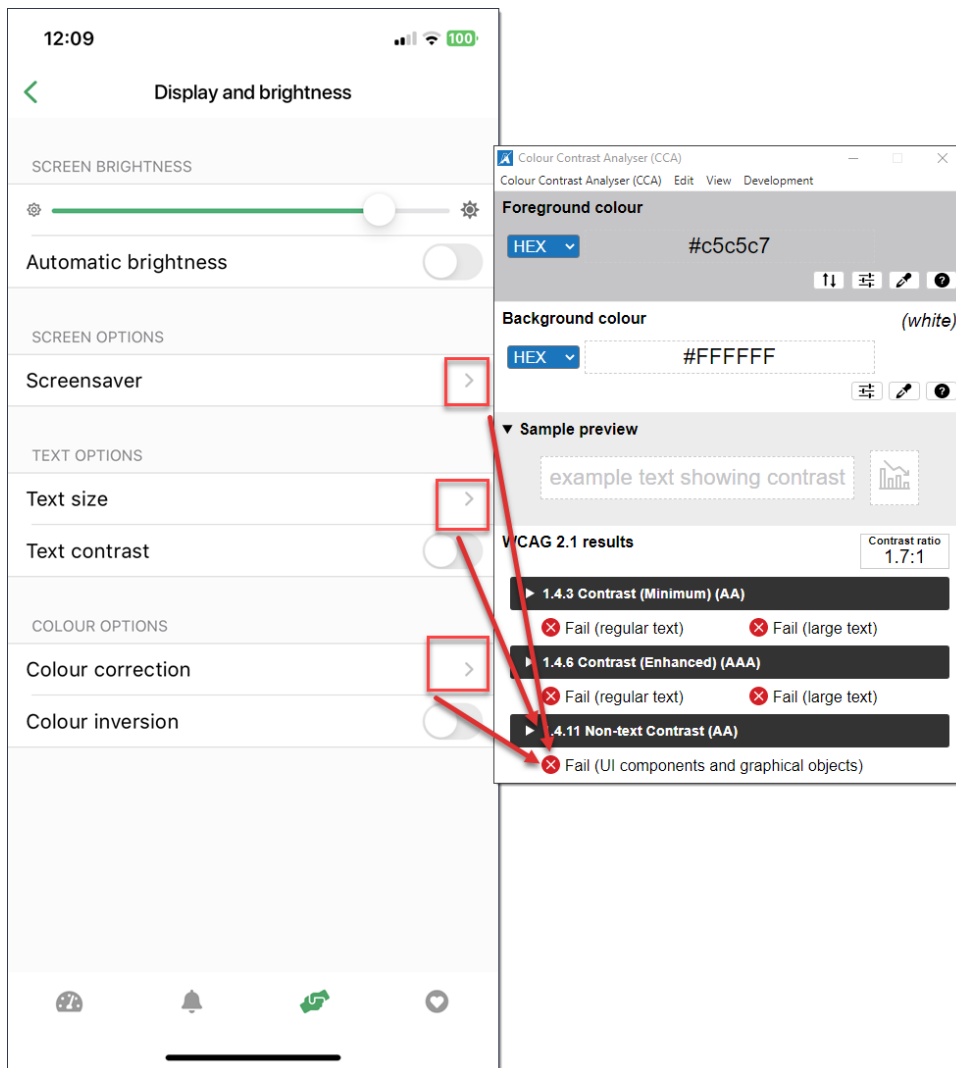
Background

The contrasts in illustrations, charts, images, icons, and videos must be sufficient. The user should be able to see and interpret the graphic information even in poor lighting conditions and with a lighter visual impairment. Therefore, strive for as good contrasts as possible. The contrast ratio in this type of material must be at least 3:1 according to WCAG. The ratio refers to the colour of the graphic motive against the background.

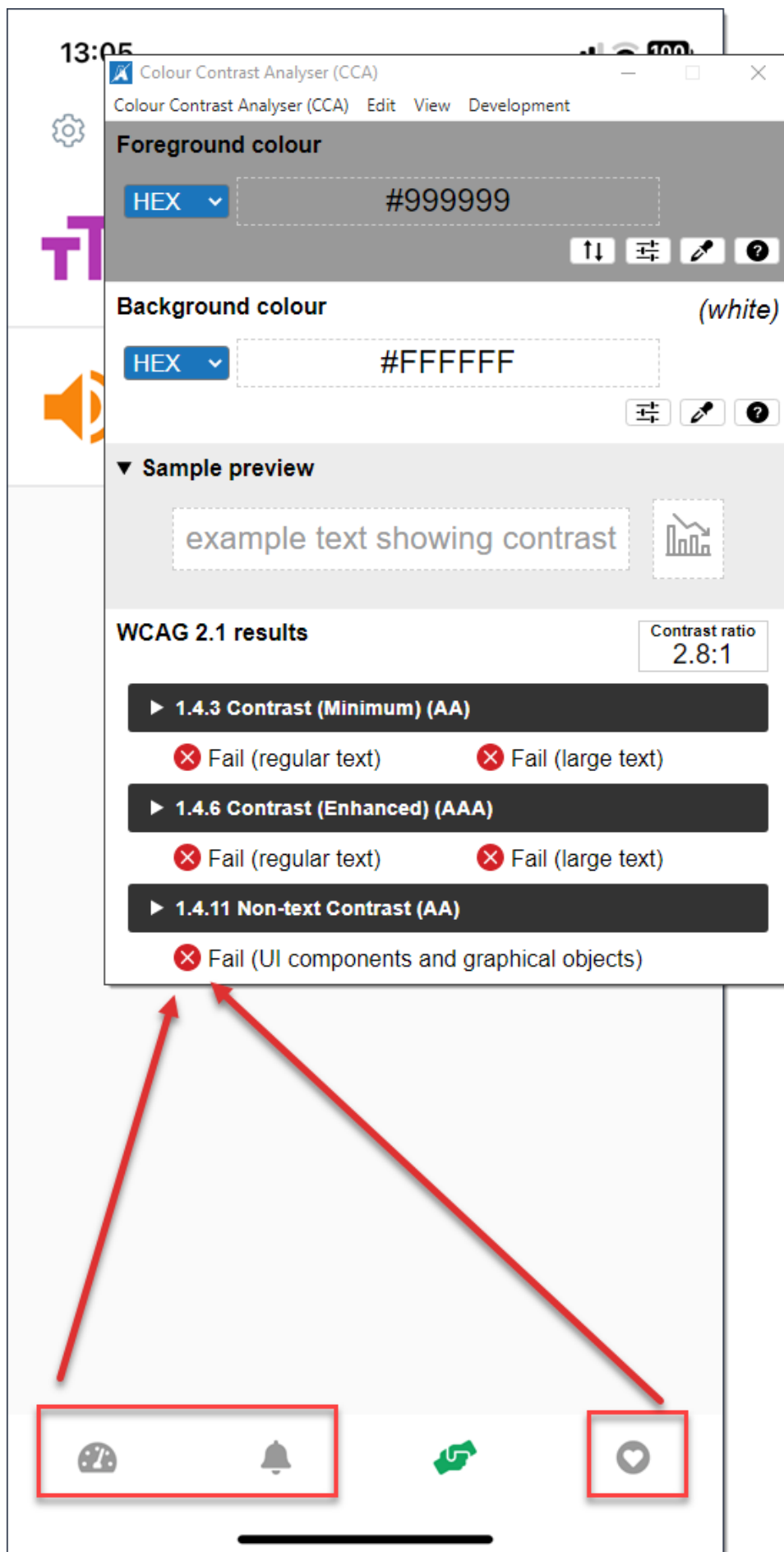
An exception to the requirement is when the object is supplemented with text.

Comment

We also found a graphical element that does not meet the minimum contrast ratio of 3:1. As shown in the example image below, the arrow is used to indicate that the area is clickable. Currently, it has a contrast ratio of 1.7:1, but it must reach at least 3:1 to meet accessibility requirements. This arrow icons is found in more then one view.



We also see that the icons that make up the menu have too little contrast against the background, which can be particularly problematic for users with impaired vision because they also lack text.



We evaluate this criterion as failed.

To do

- Increase the contrast ratio of the icons to be at least 3:1.
- Make sure all graphical objects that convey information, such as icons, have a contrast ratio of at least 3:1 in all views.

Links to guidelines

[WCAG 2.1 - 1.4.11 \(AA\)](#)

[WCAG 2.2 - 1.4.11 Non-text Contrast \(AA\)](#)

[EN 301 549 - 9.1.4.11](#)

CP25: Visual indication of clickable objects and adjacent colours have a contrast of, at least, 3.0:1

✖ Fail

Background

It is not only text that needs to have sufficient contrast and be clear in an interface. Interactive objects such as buttons, links, and form fields should be clearly distinguishable even in poor lighting conditions. In order for users with lighter visual impairments to be able to distinguish interface components, it is important that frames around such objects have good contrasts against adjacent colours. The contrast value should be at least 3.0:1.

The contrast requirement between the frame and background colour applies in several situations: when users mark a clickable object or form object, when an object is in focus when the user navigates with the keyboard, or when the user has selected a specific object.

The contrast requirement does not apply to components whose design has not been adapted, as their layout is determined by the settings of the user's browser. It also does not apply to inactive components.

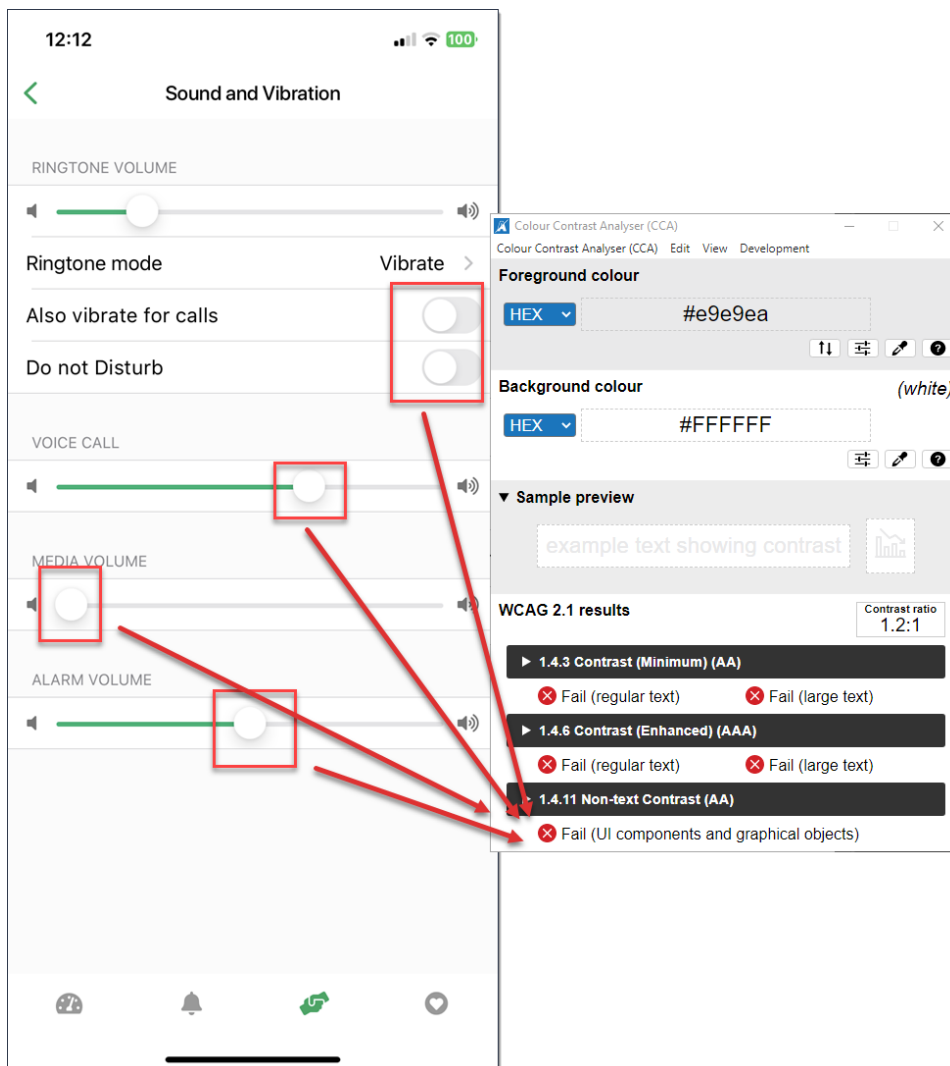
Comment

The assessment is the same regardless of app or operating system.

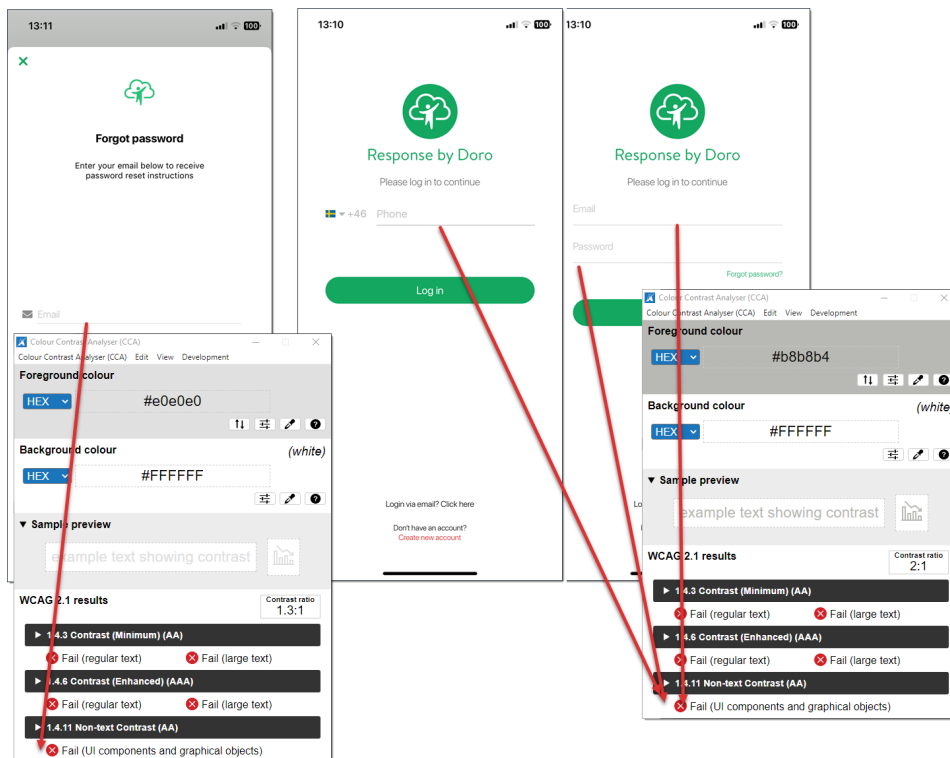
We found that clickable elements such as switch buttons and sliders do not meet the required contrast ratio. Interactive components must have a contrast ratio of at least 3:1 between the object and its background. When the switch is in the "Off" position, the contrast drops to 1.2:1, see image below, making it difficult to see for many users.

As you also can see in the example, it's difficult to see where to click on the

sliders to change the value. This becomes even more difficult when the slider is set all the way to the left ("Off").



Before we have logged in, we also see that the input fields have too weak a border. This type of border also needs a contrast ratio greater than 3:1.



We evaluate this criteria as failed.

To do

- Make sure that all clickable objects and the adjacent colors have a contrast of at least 3:1.

Links to guidelines

[WCAG 2.1 - 1.4.11 \(AA\)](#)

[WCAG 2.2 - 1.4.11 Non-text Contrast \(AA\)](#)

[EN 301 549 - 9.1.4.11](#)

CP35: Understandability is not dependent on the user's ability to perceive colour

✓ No errors found

Background

Colour can often be a clear and functional way to convey information. Red colour often indicates warning texts and error messages, blue colour is strongly associated with links.

However, you should not rely on colour alone, as many people have limited colour vision. A general estimate is that between 5% and 9% of the world's population have some form of limitation in colour vision (source: Colour Blindness Awareness, 2023).

In addition, people who does not have limited colour vision may still have problems perceiving different colours in different situations. For example, it is often difficult to perceive colour changes in running text on a mobile phone or tablet outside on a sunny day.

Make information understandable independent of colour

In order to ensure that a user's ability to perceive colour is not a deciding factor in their ability to understand the interface, you must complement with at least one more way to convey the information.

Examples:

- links can be underlined or supplemented with an icon.
- buttons in different colours can be provided with text
- diagrams can be presented with different line types.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 1.4.1 Use of Color \(A\)](#)

[EN 301 549 - 9.1.4.1](#)

CP50: The interface works well with custom settings for text

– Not applicable

Background

For users with dyslexia or visual impairments, it is important to be able to change the way texts are styled. There are various browser plugins that allow you to control the size and font, as well as the spacing between lines, words, characters and paragraphs.

It must be possible to change the text spacing without making functions inaccessible, and without hiding or cutting off the content. The result does not have to be perfect, but making changes must be possible to do in the user's browser and must not be blocked.

The user must be able to change the spacing according to these dimensions:

- Line height (line spacing) to at least 1.5 times the font size;
- Spacing following paragraphs to at least 2 times the font size.
- Letter spacing (tracking) to at least 0.12 times the font size;
- Word spacing to at least 0.16 times the font size;

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

[WCAG 2.1 - 1.4.12 \(AA\)](#)

[WCAG 2.2 - 1.4.12 Text Spacing \(AA\)](#)

[EN 301 549 - 9.1.4.12](#)

CP60: The website is presented without disturbing screen flicker, moving and flashing elements that cannot be switched off

✓ No errors found

Background

Just as a disturbing text background can be distracting, interface movements can also divert the user's attention, making it more difficult to concentrate on text. Movements, blinking and flickering should be avoided for this reason. If you wish to have this, it should either stop automatically within 5 seconds, or there should be a noticeable way to stop the movement.

Please note that anything that flashes more than 3 times per second is never allowed.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

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

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

[WCAG 2.2 - 2.2.2 Pause, Stop, Hide \(A\)](#)

[WCAG 2.2 - 2.3.1 Three Flashes or Below Threshold \(A\)](#)

[WCAG 2.2 - 2.3.2 Three Flashes \(AAA\)](#)

[EN 301 549 - 9.2.2.2](#)

[EN 301 549 - 9.2.3.1](#)

CP70: The pages have unique and relevant page titles

– Not applicable

Background

The page title is important, and is one of the first things some users meet. The page title is also used as text when the user bookmarks the page, and search engines use the page title when indexing pages.

It is therefore important that every page on your website have unique page titles that describes the page's content and function clearly.

We recommend that you first specify the unique content of the page in the title, and then the name of the website.

In HTML it might look like this:

```
<title>Funka WCAG guide - Funka</title>
```

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 2.4.2 Page Titled \(A\)](#)

[EN 301 549 - 9.2.4.2](#)

Structure & formatting

ST40: Headings reflect the content

✓ No errors found

Background

The heading is the first thing the user reads. Therefore, it is important that the heading provides relevant information about the associated text. The heading should contain one or more of the main keywords of the body text.

Both main headings and subheadings should be descriptive and concise. The heading should describe the content that follows.

If there are area headings in the page's framework, they must briefly and accurately describe area's topic. Try to use precise headings such as "E-services", "Brochures and in-depth information", or "Contact".

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 2.4.6 Headings and Labels \(AA\)](#)

[EN 301 549 - 9.2.4.6](#)

ST60: Text that visually function as a heading is coded as a heading

✓ No errors found

Background

The headings are one of the first things the user sees and reads on a new page. The importance of having good headings is therefore extremely great in order to reach out with the information. However, not all users are able to see the headings. Some users are dependent on assistive technologies to access information. In order to enable assistive technologies to understand what are headings on a page, as well as render these to the user, they need to be correctly coded with the h elements.

Several things are of importance, but a basic premise is that what looks like and, therefore, visually functions as headings are also coded with correct heading

elements.

In HTML, headings are coded with the h1, h2, h3, h4, h5 and h6 elements. The number indicates the level. It is also possible to code headings with wai-aria. Then you add the role="heading" attribute to the text and specify the level with the aria-level attribute. However, we do not recommend this since the support among assistive technologies is not 100%, and this increases the risk of problems.

There is one exception to this requirement. If you have headings that appear as soon as the page loads and are located before the content's main heading in the structure, they should not be coded as headings. An example of this is a visual heading above the left menu on a web page where the navigation is located before the main content in the structure. However, if you have a drop-down menu that only appears when the user clicks a button or link, and it has visual headings, then they should be coded as headings.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

ST80: The heading structure begins with a main heading

Background

The heading elements should form a logical structure. This means the first heading should be coded as an h1 heading.

Example:

```
<h1>Main heading</h1>
  <h2>Subheading</h2>
    <h3>Heading level 3</h3>
    <h3>Heading level 3</h3>
  <h2>Subheading</h2>
```

! No errors found but can be improved

Comment

There are techniques supported from version 15.0 in iOS to add levels to headings. It's missing in the app, but since it is possible to add it, we recommend that you review that possibility.

We assess this checkpoint with no errors found but can be improved.

To do

- Review the possibility of adding heading levels and that the heading structure begins with a main heading.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

ST90: The heading structure is logical and represents the hierarchy of the content

! No errors found but can be improved

Background

All pages should have a main heading that provides the user with information about the text and gives an opportunity to the user to choose whether to start reading or not. Any text longer than a couple of paragraphs should also have subheadings. Subheadings are usually required to make the text comprehensible and clear, both for experienced and inexperienced readers. They can also be of great help to people with reading difficulties.

Subheadings also facilitate skim-reading, especially for people who use reading aids. Many aids present the headings as a table of contents of the text, and also make it possible to jump between headings. In this way, the user does not have to read through the entire text to reach the information of interest.

The heading structure must be logical and reflect the structure of the page's content. Begin all pages with a main heading (the h1 element) at the top of the content. Subsequently, all headings should come in a hierarchically correct order that reflects the structure of the content. Keep in mind that the number should not reflect how important the heading is, but how it relates to other headings. Is it a subheading to the previous heading, or should it be on the same level?

Comment

As we mention under ST80, it's possible from version 15.0 in iOS to add levels to headings. As it's missing in the app, this checkpoint is also assessed with no errors found but can be improved.

To do

- Review the possibility of adding heading levels. Then make sure that headings are structured according to how they relate to each other and that heading levels are not skipped.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

ST120: Lists are correctly coded and used in a correct manner

✓ No errors found

Background

Some information needs to be entered into lists. This includes enumerations, definition lists, and descriptions of different steps in a process.

To insert these as lists in the code, the elements available in HTML for this purpose should be used. This is a prerequisite for assistive technologies to be able to interpret and present them as correct lists to the user.

To start a bulleted list the element `ul` is used, to start a numbered list the element `ol` is used. Each list element is then created with the element `li`.

Lists can also be used to structure the menu. By putting the menu in a nested list, you can convey the hierarchy of the menu in a basic way even to users who cannot see it.

Example:

```
<ul>
  <li><a href="b.html">Breakfast</a></li>
  <li><a href="l.html">Lunch</a>
    <ul>
      <li><a href="m.html">Meat</a></li>
```

```
        <li><a href="v.html">Vegetarian</a></li>
    </ul>
</li>
    <li><a href="d.html">Dinner</a></li>
</ul>
```

For definition lists, the dl element is used to start the list, the dt element to indicate the term to be defined, and the dd element to provide its description.

Example:

```
<dl>
    <dt>Sweden</dt>
    <dd>Stockholm</dd>
    <dt>Norway</dt>
    <dd>Oslo</dd>
</dl>
```

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

ST180: Paragraphs are created correctly using the p element

× Fail

Background

To insert text paragraphs on a page, the p element should be used. This makes the document easier to skim-read with assistive technologies. The user can then jump from paragraph to paragraph. This is especially important in legal texts where the different paragraphs must be distinguishable from each other.

Do not use the br or hr elements to separate text paragraphs.

Comment

The assessment is the same regardless of app or operating system.

There isn't much text in the app, except under Settings/Help. Here, we found that it is very difficult for screen reader users to access the content under "Terms and Conditions" and this is because the visual text appears as one large text block.

Screen reader users typically navigate from block to block and focus on one block at a time, but this isn't possible in the current setup. As a result, reading this kind of content becomes very time-consuming.

No functionality in the app is affected by this issue.

The example image below shows the view.



Terms and conditions

Territory of Sale

1. Limitation of service and necessary equipment

The following terms and conditions constitute the terms of the **Agreement** between you as a User (hereinafter referred to as "**User**") and Doro AB (hereinafter "**Doro**") a company registered in Sweden, with company registration number 556161-9429, with regards to the free of charge service **Response by Doro** (hereinafter referred to as the "**Service**").

In terms of this Agreement there are two types of Users and unless addressed specifically as per below, the term "**User**" will be used in general for both:

1. **Responders**, i.e. the parties that will receive the alarm signals from the senior (hereinafter referred to as "**Responder**"). These are often close relatives of the Senior.
2. The **Senior**, i.e. the party that will have a Doro device and activate the alarm signal from the Doro device in case of emergencies (hereinafter referred to as "**Senior**").

Response by Doro is a built-in free of charge service on the Doro devices providing the Senior with a Response button on their Doro device, which, when pressed, will send a distress alarm signal to all Responders simultaneously. Response by Doro also provides Responders (e.g. relatives, friends and family) with the ability to remotely assist a Senior (i.e. the Doro device user) with basic settings like adjusting sounds, screen brightness etc. To make full use of this service Seniors will need to share location details and by signing up to this service you acknowledge that responders are allowed to identify the Users location. The Response by Doro service is free of charge, and available in all markets where the Doro senior devices are sold.

Doro Software: In order to provide the Response by Doro service, Doro is making its Response by Doro software (hereinafter "Doro Software") available to the User. By downloading, installing or otherwise using the Doro Software, the User agrees to be bound by this Agreement. The User must make sure to have fully read and understood this Agreement before using Response by Doro as well as the Doro Software.

Territory of Sale, the territory where the Response by Doro Service and the Doro devices are sold.

Usage abroad. Doro has not designed the service to be used, and is not responsible for, use outside of the Territory of Sale and does not guarantee the Service outside the Territory of Sale.

No rights or obligations under this Agreement or any part thereof may be transferred to third parties without Doro's written consent.

We evaluate this criteria as failed.

To do

- Ensure that large blocks of text are divided into smaller paragraphs to improve readability and make navigation easier for users.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

ST200: Quotes are marked up using q or blockquote

– Not applicable

Background

Mark up longer quotes with the blockquote element.

Note that blockquote must be used only for quotes. Visually, the element creates an indent in the text, but do not use the element only to create visual indents.

Use the q element to highlight and format shorter quotes.

In both blockquote and q elements, the cite attribute should be used to indicate a source in the form of a URL if such a source exists.

Example:

```
<blockquote  
  cite="http://www.bbc.com/weather/prognosis_071224.html">  
  
  <p>There is a 50-50 chance of snow in northern Europe for  
  Christmas. However, this is based on the low pressure system  
  now over the British Isles steering northwards.</p>  
  
</blockquote>
```

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

Images

IM10: Text is presented as text, not as images of text

✓ No errors found

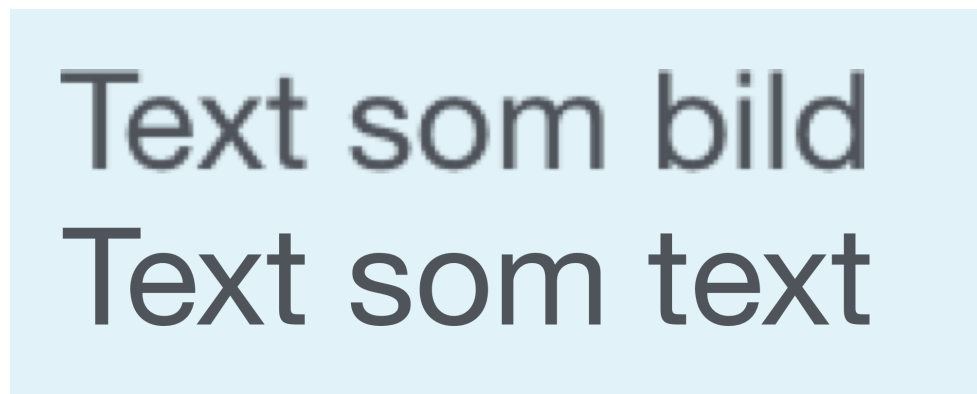
Background

Images are, in many ways, a better way of communicating information than plain text. Images can help illustrate complex connections and difficult concepts. Images also help in that they break up the text and make the user experience the text less intense and difficult.

Images of text

In other words, images should be used where appropriate, but they should not be used to represent text. When text is used in the content, in menus, or other parts of the pages on the website, it should be added as plain text. Images of text usually create large accessibility problems.

When images are zoomed, each image pixel becomes larger. Text, on the other hand, is inserted as vectors, which means that each character is made up of more pixels when the text is zoomed. The difference is that the image of text becomes pixelated and is often perceived as blurrier, while the normal text remains sharp. This becomes especially apparent in magnifying tools. The two images below illustrate this. The first represents an image of text which is created with pixel graphics, and the second is an image of real text which is created with vector graphics. It's the same text size, and we've zoomed in 300%.



The image shows two lines of text, 'Text som bild' and 'Text som text', on a light blue background. The first line, 'Text som bild', is rendered in a pixelated font style, where each character is composed of distinct, blocky pixels. The second line, 'Text som text', is rendered in a smooth, vector-style font, where the characters are continuous and sharp. Both lines of text are in a dark gray color and are centered horizontally. The background is a solid light blue.

Another problem with images of text is that the text in an image cannot be

adapted by tools for dyslexics, nor can it be copied, and that is something some of the assistive tools for this group require. Even if you add an alt text to the image, it is not certain that a tool for dyslexics can perceive this.

Exceptions

In certain situations, there can be reasons to use text in an image, for example, in organization charts or different types of flowcharts where the text needs to be part of the image.

When text is used in such images, we recommend it to be at least twice the size of the normal text on the website (as the text in the image cannot be zoomed in a good way by browsers and assistive tools). At the same time, the contrasts must be good, and the image must contain an alt text.

If the text cannot communicate all the information the image shows, there must either be a detailed description inserted via a detailed text description or a link next to the image. If it is an organization chart, the organization's different departments can be described in text under the image. Use headings and lists to structure the information as clearly as possible.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 1.4.5 Images of Text \(AA\)](#)

[WCAG 2.2 - 1.4.9 Images of Text \(No Exception\) \(AAA\)](#)

[EN 301 549 - 9.1.4.5](#)

IM30: Equivalent text descriptions are present for all meaningful graphical elements on the website

× Fail

Background

In order for assistive tools to render the content of images to the user, the images must be provided with an alternative description. This is usually done using the alt attribute (which can be called a tooltip or description in various authoring tools).

The alternative description should briefly describe the subject or purpose of the image. The more important the image is for the page and the understanding, the more detailed the description needs to be.

The alt text should be kept to less than 150 characters, but sometimes this is not enough to describe all the information in the image. Then the alt text must briefly state what the image shows and refer to a longer description or corresponding information. For example:

"Graph of population growth in Sweden. See table after the image".

As a principle, all images that contain information, illustrations, photographs, icons, image buttons, image links, and similar, are meaningful graphic objects. Non-meaningful graphic objects include hyphens, empty spaces, coloured background tiles, and similar. These kinds of images should primarily be inserted with CSS.

Linked images require a little extra thought. If you have a link that only consists of an image, the image's alt text must primarily reflect the target of the link. If you have a link that contains both an image and a text, then the link description will consist of the link text and the image's alt text. Depending on the situation, you may need to have an empty alt text. For example, it's just annoying to hear:

Printer Print, if you have a print link with an icon.

If you have a form and use the input type="image", this must have an alt text that explains the function of the object. However, note that it is usually better to use input types that are adapted for the purpose, for example, it is better to use the input type="submit" or a button element for buttons, rather than an image.

If you use the figure and figcaption elements, the image must have an alt text.

Example:

```
<figure>
  <img ... alt="Sweden year 1790" />
```

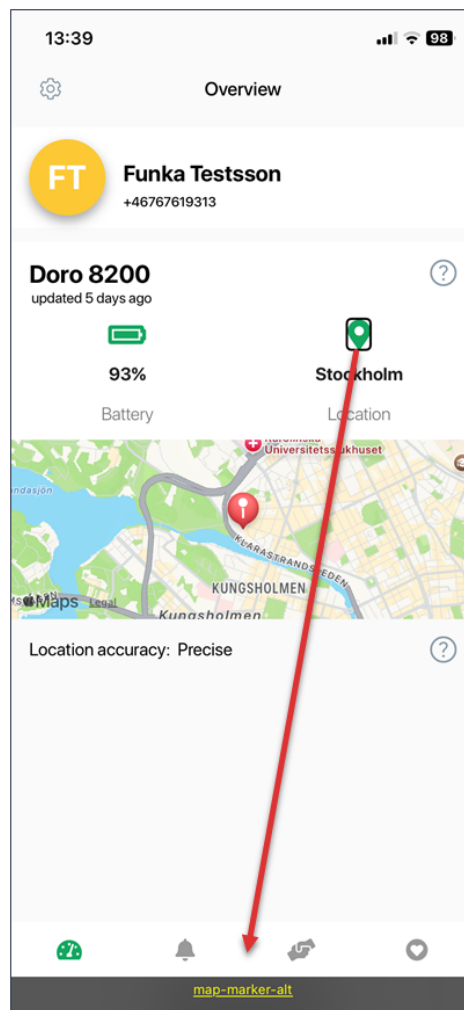
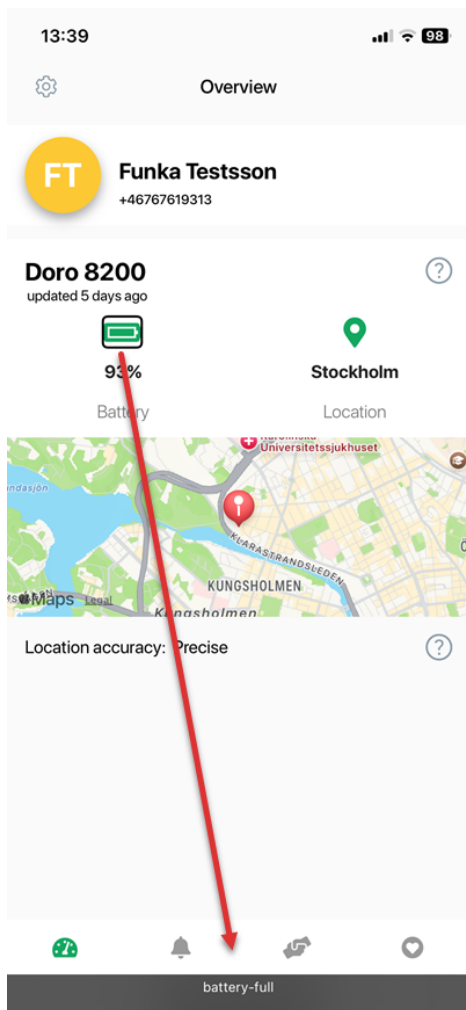
```
<figcaption>
  The image shows Sweden's distribution in 1790...
</figcaption>
</figure>
```

Comment

The app contains only a few images, which we consider good practice for this type of interface. Meaningful graphical elements should be focusable and include descriptive labels. If an image is purely decorative or simply complements text, it should not be focusable.

In this case, the images next to "Battery" and "Location" are not essential, they serve only as visual complements to the text. Therefore, they should not be described by screen readers, nor should they be focusable.

Currently, the images have labels such as "battery full" and "map marker alt", which is misleading, see image below. For example, a screen reader user might wrongly assume that the battery is actually full, which could cause confusion.



We evaluate this criteria as failed.

To do

- Make sure that non-meaningful images and icons is hidden and not focusable.

Links to guidelines

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

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

[WCAG 2.2 - 1.1.1 Non-text Content \(A\)](#)

[EN 301 549 - 9.1.1.1](#)

IM60: Image maps have text descriptions for both links and images

! No errors found but can be improved

Background

Image maps are images that contain clickable areas. There are two types of image maps: client-based and server-based.

A client-based image map is a regular image where parts of the image are defined as linked areas.

A server-based image map is an image where you can click anywhere. The coordinates are sent to the server, which in turn determines what is sent back to the browser. This technology is used in many zoomable map services such as Google Maps.

Accessible client-based image maps

For users with assistive technologies, client-based image maps work best. It is relatively easy to ensure that all areas have an appropriate description.

The image used for the image map should have a brief description of what the image depicts, just like any meaningful image.

The different clickable areas (area elements) must be described in the title attribute, but can also be complemented with a description in the alt attribute.

You can also complement with plain text links to the same pages that the links in the image lead to.

This is what the code for a simple client-based map might look like. Here we have chosen to use both alt text and title text in the clickable areas, because the support for both attributes varies in the area element:

```

<map id="map" name="Map">

  <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="See
  projects in northern Europe" title="See projects in northern
  Europe">

  <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="See projects in central Europe" title="See projects in central Europe">
```

```
<area shape="poly"
coords="15,165,27,180,80,192,140,204,138,247,18,249,4,171"
href="www.funka.com" alt="See projects in southern Europe"
title="See projects in southern Europe">
</map>
```

Comment

The assessment is the same on iOS as on Android.

We see that you are using server-based maps from Google where there are accessibility issues. These accessibility issues include, among others, contrast issues and drag movements. However, as the map is a supplement to a text, we do not consider the criteria to be failed.

16:07



Overview



Funka Testsson

+46767619313

Doro 8200

updated Just now



43%

Battery



Stockholm

Location



Location accuracy: Precise



Nybrokajen 7



SE-111 48 Stockholm +46 8555 770 60



contact@funka.com



www.funka.com/en

Funka

We evaluate this criteria as no errors found, but can be improved.

To do

- Strive to make your server-based maps as accessible as possible.

Links to guidelines

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

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

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

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

[WCAG 2.2 - 1.1.1 Non-text Content \(A\)](#)

[WCAG 2.2 - 2.4.4 Link Purpose \(In Context\) \(A\)](#)

[WCAG 2.2 - 2.4.9 Link Purpose \(Link Only\) \(AAA\)](#)

[EN 301 549 - 9.1.1.1](#)

[EN 301 549 - 9.2.4.4](#)

Forms

FS10: Forms are coded correctly with form elements

× Fail

Background

It is important that all parts of a form are inserted as correct objects designated for forms, otherwise, they will not be presented together with other parts of the form when being presented through an assistive tool. In some cases, buttons appear to be created as linked images (A elements) instead of correct buttons in the form. When this is the case, they will not be listed with other objects in the form, and this makes it difficult for some users to submit the form.

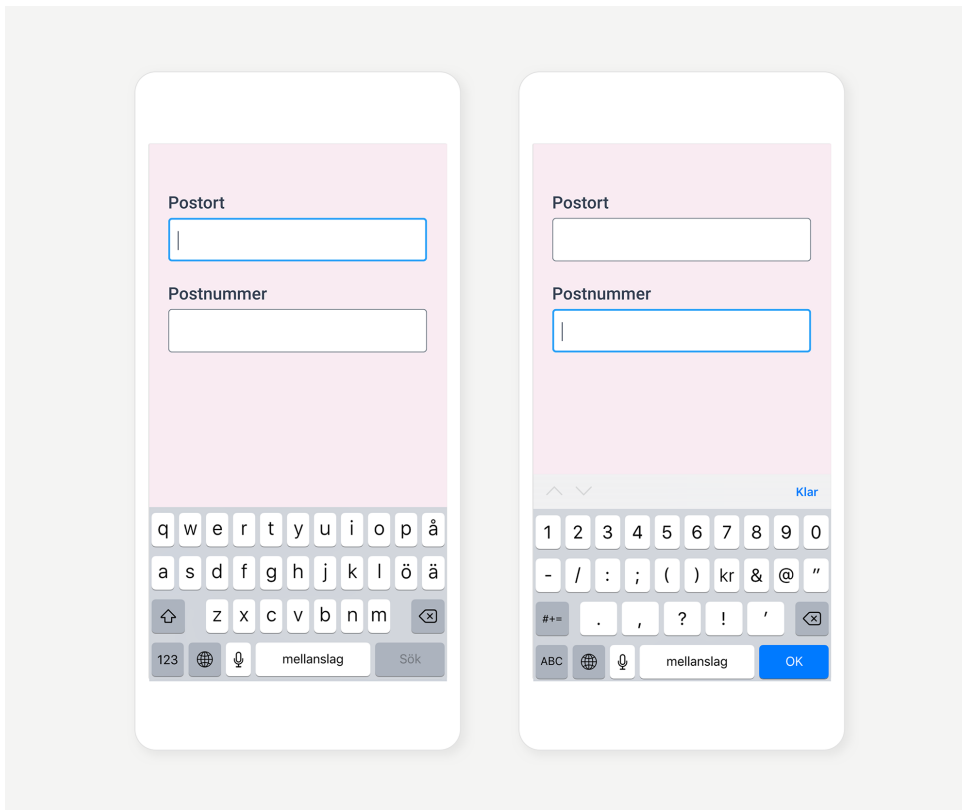
It is also important that the right type of element is used. Instead of using, for instance, checkboxes, and adding a script that prevents anything other than one checkbox from being checked at a time, radio buttons should be used.

There are several new types of objects designated for forms in HTML5. These have many advantages for users, especially on mobile devices where the keyboard can adapt to the type of information that is to be entered. An example of how different input types generate different keyboards on mobile is shown below.

```
<label for="postort">postort</label>
<input type="number" id="postort">
```



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



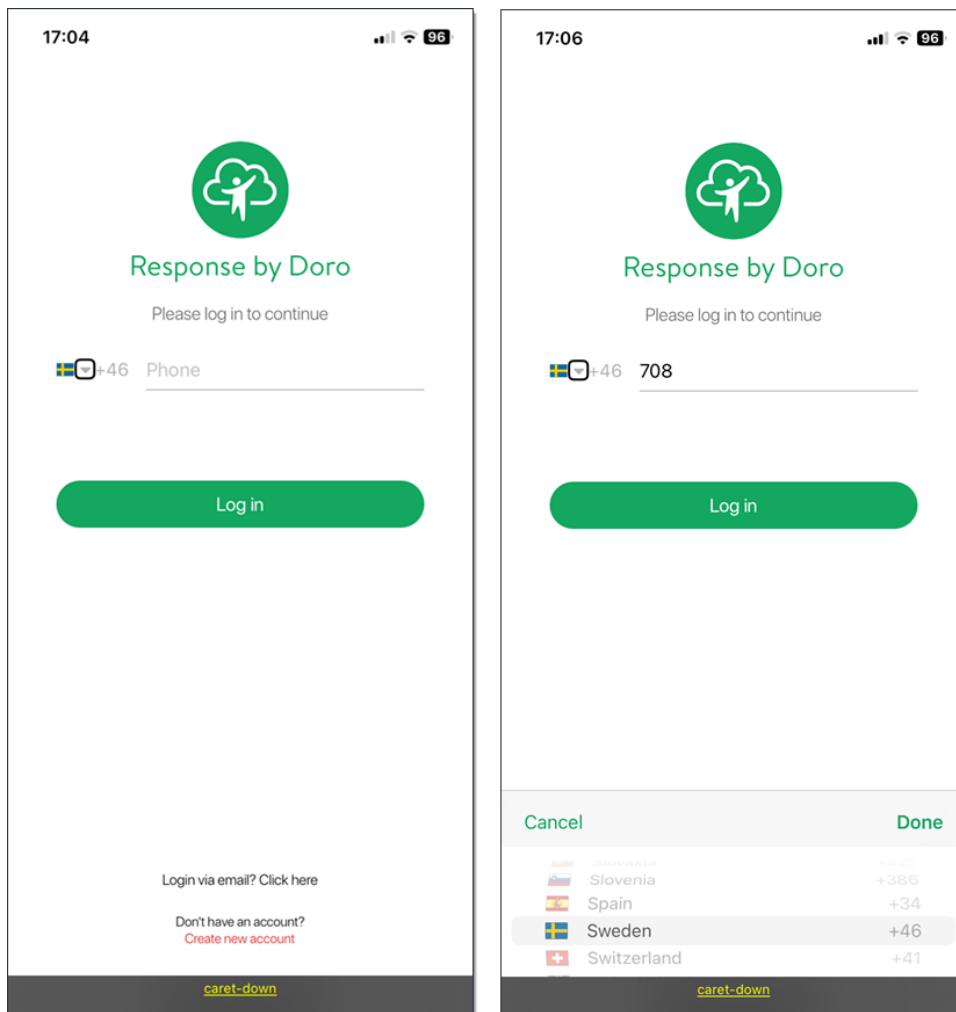
Comment

In the startup view, we found a dropdown component with serious accessibility flaws, as it lacks both a proper role and an understandable name.

With a screen reader, it is possible to focus on the small icon with a downward arrow, and the only information announced is "caret down", see image below. If the user tries to activate the element, a "scroll wheel" is visually displayed, but no information is conveyed to the screen reader, the interaction is purely visual.

Therefore, screen reader users are not informed that they can select a country using this component. If the component had a proper name, it might be possible to understand that it is intended for selecting the country code.

Most likely, users using screen readers will not be able to log in.



We evaluate this criteria as failed.

To do

- Ensure that dropdowns works as they should even when screen reader.
- Make sure that all form elements have a name, a correct role and a value.

Links to guidelines

[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 \(A\)](#)

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[WCAG 2.2 - 4.1.2 Name, Role, Value \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

[EN 301 549 - 9.4.1.2](#)

FS50: There are labeled instructions that describe how to complete forms

✓ No errors found

Background

When the user chooses or enters information, there must be clues and instructions that clearly indicate what to choose or enter. The instructions should also contain information about specific formats and other important information that the user needs in order to enter information correctly.

Mandatory fields

In some cases, the users must fill certain inputs to complete a form. It must be clearly distinguishable which fields are mandatory and which fields are not. To only use an asterisk (*) does not indicate for all users that an input field is mandatory - it should appear in text what the icon means. At Funka, we recommend that if a majority of the input fields are mandatory, you should instead write in which that are voluntarily filled, please see the example below.

Deltagarinformation
Namn

Titel/ Arbetsområde (valfri)

E-post

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 3.3.2 Labels or Instructions \(A\)](#)

[EN 301 549 - 9.3.3.2](#)

FS60: Labels are linked to corresponding form object



Background

Instructional text is a text that states what the user should enter in a certain field, the significance of checkboxes/radio buttons, or what the user should choose in a list. These instructional texts are often clearly positioned visually close to the form object (text field/checkbox/radio button/list/button) in the question, which means users who can see the page and read the text are clearly informed of the form object's purpose.

In order for this to work for users who need various tools to access the information, there must be a clear connection in the code between the instructional text and the form object. This is achieved with the label element. Use label-element to insert all instructional texts. The label element must also be correctly connected to the respective form object with the attribute for that pointing to the form object ID.

In addition to the instructional text, the label element should also include possible clarification and a possible error message bound to the form object. Please note that you should only have 1 label element for each form object.

If the interface is coded in HTML5 the form field should be labeled with what type of field it concerns.

```
<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>
```

This code will generate following form:

Namn

Land

 ☐

Ja, jag godkänner

☐

Nej, jag godkänner inte

Kommentar

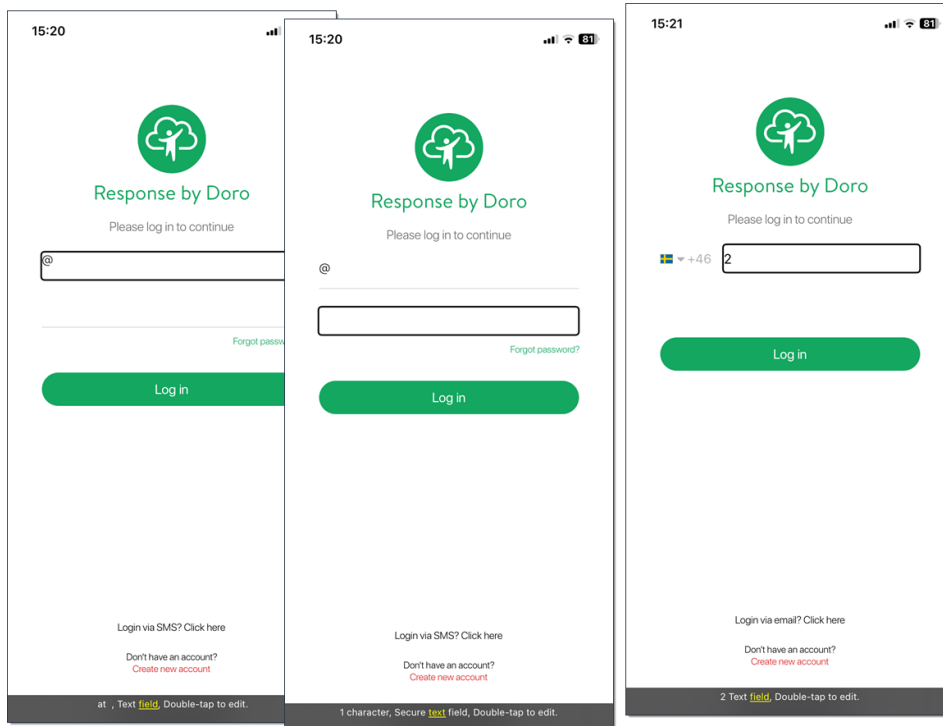
Skicka svar

Comment

The assessment is the same regardless of app or operating system.

We identified accessibility issues with the form fields for the country code and mobile number. At login, the input field for the mobile number lacks a label that indicates what should be entered. This problem is the same when we try the fields in the view to log in with email and password but only for iOS, on Android it worked fine.

All form fields must have a clear label and an accessibility name. This label must also not disappear when the user starts typing in the field.



We evaluate this criteria as failed.

To do

- Ensure that every form field in the app has a visible label for sighted users and an accessibility name for screen reader users. They should also be the same.
- Also make sure that a label is visible at all times and does not disappear if the user starts typing in the input field.

Links to guidelines

[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 \(A\)](#)

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[WCAG 2.2 - 2.4.6 Headings and Labels \(AA\)](#)

[WCAG 2.2 - 3.3.2 Labels or Instructions \(A\)](#)

[WCAG 2.2 - 4.1.2 Name, Role, Value \(A\)](#)

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: Input fields in forms are identifiable through correct use of the autocomplete attribute

– Not applicable

Background

Forms are a major problem for many users. This is particularly true for long and complex forms, and especially for users with cognitive disabilities. The simpler the form and the less the user has to fill in, the less likely they are to have problems. Many other groups also have particular problems with forms. Examples include users with writing difficulties and users with severe visual impairment.

The autocomplete attribute makes it easier to fill in form fields that are intended to contain frequently recurring information, such as name and email address. The user can enter the information once and let the browser save it for the next time the user enters the same information in other form fields. Autocomplete is used on form fields of different types such as text, email, number, password.

```
<form action="/subscribe.html">
  Given name:<input type="text" name="gname"
    autocomplete="given-name" />
  Family name: <input type="text" name="fname"
    autocomplete="family-name" />
  E-mail: <input type="email" name="epost" autocomplete="email"
    />
  <input type="submit" value="Subscribe" />
</form>
```

There are about 50 possible values to use in the autocomplete attribute.

In some situations it is not desirable for the browser to complete the form fields. Examples of such situations are when the user has to enter a one-time code when logging in to an e-service. In this case, the automatic completion can be turned off by setting the autocomplete attribute to "off" on specific form fields.

Assistive devices for people with cognitive disabilities can detect the autocomplete attribute in form fields and provide the user with a simpler and more

understandable description of the information to be included in the form field. An example is when autocomplete is set to the value "tel" which replaces or supplements the form field's lead text with a phone icon.

To meet the requirement of this point, form fields listed in the W3C "Input Purposes for User Interface Components" list should have the autocomplete attribute set to the values specified in the list.

[Input Purposes for User Interface Components](#)

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

[WCAG 2.2 - 1.3.5 Identify Input Purpose \(AA\)](#)

[EN 301 549 - 9.1.3.5](#)

FS80: Extended descriptions of a particular form field is tied to that specific field

– Not applicable

Background

It is not always enough to have a common label text to explain to the user what, how, and why certain information in the form is required. When the need for an extended explanation arises, for example, to clarify the date formats, or why you ask the user to enter certain information, the explanation should be linked to the current form object.

The easiest way is to let the extended description be part of the label element. This is the most robust method. You can also use the wai-aria attributes aria-labelledby and aria-describedby, however, this will cause some users to miss the information. Whether this is a problem or not, it must be sorted out on a case-by-case basis since it depends on the situation and on which users the system is addressing.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

FS85: The user does not need to fill in the same information more than once per visit

– Not applicable

Background

Some users with cognitive disabilities may have difficulty remembering what they entered previously.

For example, each time the user uses a service to schedule their medical appointments, the user must rewrite certain information that the user provided in a previous step.

Information that has been previously entered or provided by the user and is required to be entered again in the same process is either: automatically filled in, or available for the user to select.

Valid exceptions apply when it is:

- significantly important to enter the information again,
- the information is required to ensure security,
- or previously provided information is no longer valid.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

[WCAG 2.2 - 3.3.7 Redundant Entry \(A\)](#)

FS100: The user can review, undo, or confirm information while performing important tasks

– Not applicable

Background

It is common to perform important tasks digitally, such as transferring money or signing a contract. If the user accidentally makes a mistake, such as pressing the wrong button or misunderstanding information, the consequences can be significant.

Therefore, the user must be provided with assistance to ensure they do things correctly and avoid unnecessary problems. The best approach is to offer the user multiple safeguards. For example, they may be able to review a summary and confirm the information before it is sent, and have the option to undo the action.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 3.3.4 Error Prevention \(Legal, Financial, Data\) \(AA\)](#)

[EN 301 549 - 9.3.3.4](#)

FS120: The function of buttons is clear

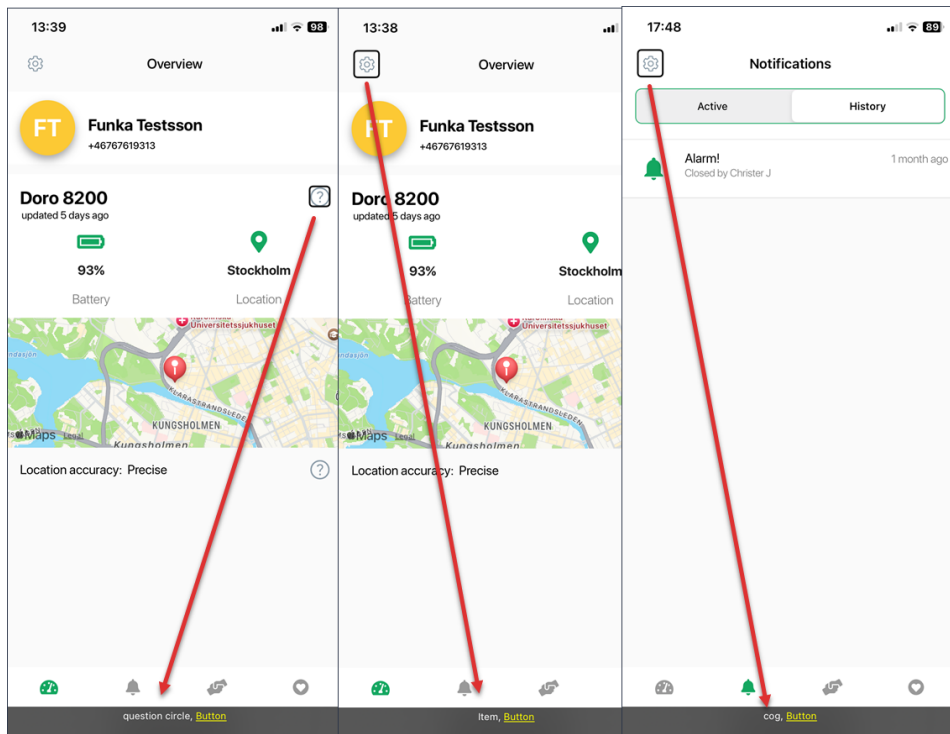
× Fail

Background

It is important that users understand what will happen when they click on a button. Therefore, buttons ought to be self-explanatory so the user understands their purpose. For example, a button named "OK" is not as obvious as a button named "Comment" or "Subscribe".

Comment

Buttons should be easily recognizable, have the correct role as a button, and also has a programmatically determined name so that screen reader users can understand their purpose. We found some buttons that are visually presented as buttons but have unclear or incorrect names, see examples below.



16:20



Responders



SENIOR NETWORK



PENDING

This is the Group of Responders for Funka Testsson who will receive alarms.



Mia Testsson (you) ✓

+46767619329



Christer Janzon ✓

+46708231065



INVITE



We evaluate this criteria as failed.

To do

- Make sure that buttons have a proper name that is possible to understand.

Links to guidelines

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

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

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

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

[WCAG 2.2 - 1.1.1 Non-text Content \(A\)](#)

[WCAG 2.2 - 4.1.2 Name, Role, Value \(A\)](#)

[EN 301 549 - 9.1.1.1](#)

[EN 301 549 - 9.4.1.2](#)

FS130: Sections of forms are grouped

– Not applicable

Background

Longer forms must be split and grouped. This makes it easier to see which parts belong together, both visually and with assistive technology. Grouping objects is done with fieldset and legend elements.

Example of a form with groupings:

```
<fieldset>
  <legend>Delivery address</legend>

  <label for="name1">Name:</label>
  <input type="text" name="name1" id="name1">

  <label for="adress1">Address:</label>
  <input type="text" name="address1" id="address1">

  <label for="ort1">City:</label>
  <input type="text" name="city1" id="city1">
</fieldset>

<fieldset>
  <legend>Billing address</legend>
```

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

<label for="adress2">Address:</label>
<input type="text" name="address2" id="address2">

<label for="ort2">City:</label>
<input type="text" name="city2" id="city2">
</fieldset>
```

As soon as you have a group with radio buttons or checkboxes, you should use a fieldset and legend to give the group a common description. Without this, you could say in very simplified terms, it is like asking the user to answer a question without hearing the question first.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

FS140: When an error is made, the user is made aware of this in a clear way

✓ No errors found

Background

If there is a possibility for users to make an error, errors most likely will occur. If this happens, it is important that the user is notified that something has gone wrong.

Provide a combined error message at the top

There are several different ways of showing that an error has occurred, however, to ensure as many users as possible understand this, most forms should offer a combined error message uppermost. This should clearly state:

- An error has occurred. We recommend using both a heading and an icon to state that it is an error message.
- Which errors have occurred? Ideally in a bullet list.
- Where the errors have occurred. Name the form objects where the errors can be found.
- If possible, also what the user should do to correct the errors.

Describe the error by the respective field

Provide clear and distinct error messages next to the invalid form fields, preferably combined with an icon. The error should also be described to the user here.

What more can I do?

It would be even better if you offer live validation so errors are detected while the user is completing the field. However, keep in mind that not all users will see this; it must therefore also be possible to submit the form even if the user has not corrected the errors. When the user has submitted the form, a combined error message as above should be shown.

It may also be worth linking each error message to the corresponding form field so that users do not have to search for them.

Other types of error

Another example of errors that can arise is when the user tries to load a page that does not exist. Thus, the website must provide a clear error message that states that the page in question does not exist and give the user an opportunity to find the information in another way.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 3.3.1 Error Identification \(A\)](#)

[EN 301 549 - 9.3.3.1](#)

FS150: Error messages are tied to their respective form field



Background

When error messages are inserted in a form and relate to a specific form object or a group with objects (for example a group of radio buttons), these must be tied to the form object or group. This is normally done by inserting the error message as part of the label or legend element.

This can possibly require that the error message is embedded twice, first to be displayed visually and second to be hidden in the existing instructional text. In this situation, it is advisable to conceal the visual error message for the tool and make sure the visual concealed error message is read. This can be done like this, for example:

HTML-code:

```
<label for="name">Name: <span class="concealed">Fail: You  
have omitted to enter your name</span></label>
```

```
<input id="name" type="text">  
<span aria-hidden="true">You have omitted to enter your  
name</span>
```

CSS-code:

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

Naturally, it is preferred to enter the error message only once and visible for all users.

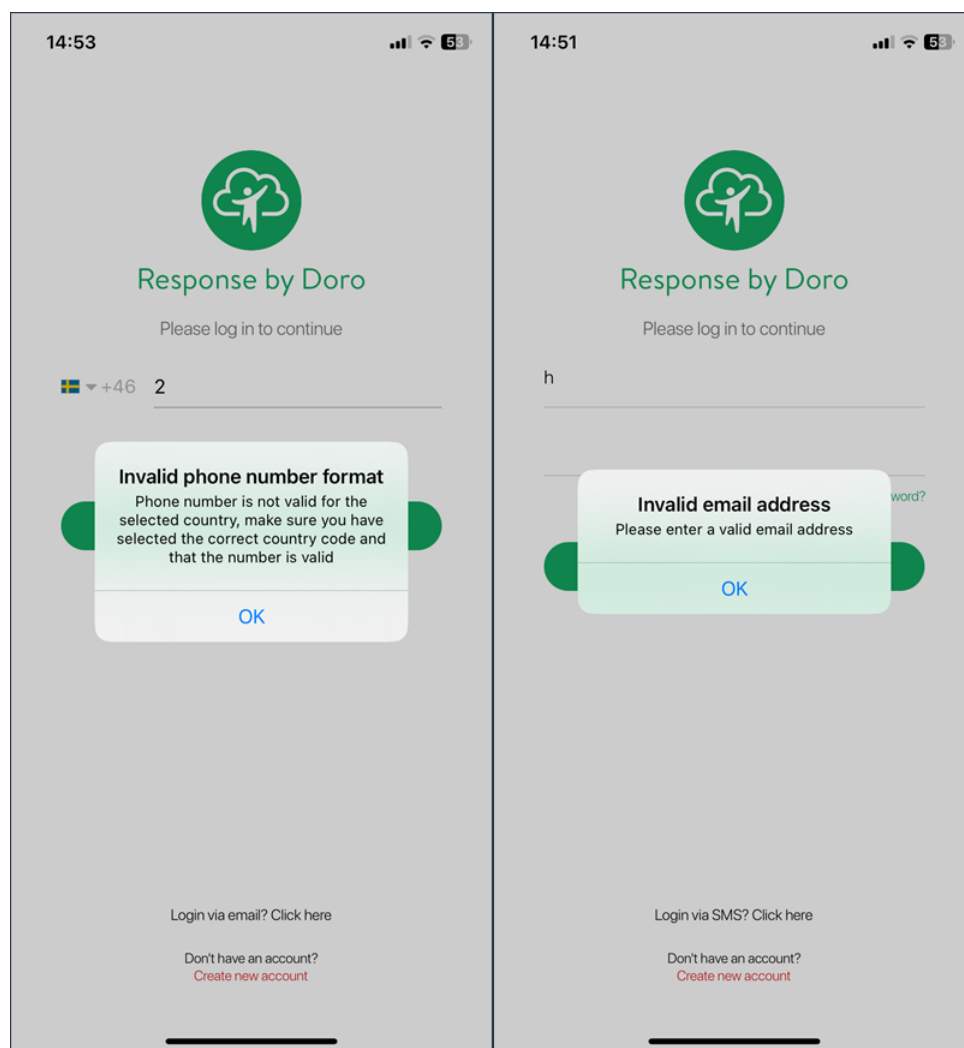
Comment

We find that the error messages are clear and focus is correctly placed on the warning messages when they appear. However, the error messages are not programmatically associated with the corresponding input fields, they are presented as a general alert message instead.

Ideally, when a user focuses on a form field that contains an error, the error message should be read out by the screen reader.

This becomes even more difficult because the input fields lack accessible names,

making it hard for screen reader users to know which field the error refers to.



We evaluate this criteria as failed.

To do

- Make sure that error messages is tied to the form field so that screen readers also receive this information.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

FS170: All errors are described with text

✓ No errors found

Background

Error messages may contain pictures or even movies, but it is important that the error message is presented as text. Therefore, it is not enough to use an icon to highlight that something is wrong, this should always be explicitly stated in text form.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 3.3.1 Error Identification \(A\)](#)

[EN 301 549 - 9.3.3.1](#)

FS180: When possible, an explanation on how to rectify errors is provided

✓ No errors found

Background

Describe in as much detail as possible what the error is, and how the user can fix it. For example, you can give suggestions on what might be wrong, like this:

You must enter an email address that contains an @ character

This is better than "Enter a valid email address", which is unclear.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 3.3.3 Error Suggestion \(AA\)](#)

[EN 301 549 - 9.3.3.3](#)

Tables

TB40: Table data is correctly coded with table code

– Not applicable

Background

Ideally, you should always code tables with the correct table code.

CSS can be used to create tables that look like tables but are not coded as proper tables. There are also html5 roles and wai-aria attributes that can be used instead of traditional html code to create tables. However, these solutions do not always function properly for users with assistive technology.

For this reason, you should aim to always code tables with the elements table, tr, th and td.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

TB50: Table headings are coded with the caption element

– Not applicable

Background

In order for users to understand a table easily, there must be clear table headings that are inserted correctly.

Use the caption element. It appears visually as a caption and at the same time gives the table a name. The name allows assistive technology to distinguish between different tables. The caption should briefly explain what information the table contains.

The caption element is placed at the top of the table at the same level as table rows (tr elements):

```
<table>
```

```
<caption>Flight prices Stockholm - Oslo</caption>
```

```
<tr>
```

```
...
```

The image below shows an example of a table header.

Flight prices Stockholm - Oslo		
MORNING FLIGHTS	AFTERNOON FLIGHTS	EVENING FLIGHTS
1500:-	1750:-	1750:-
150€	175€	175€
150\$	175\$	175\$

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

TB60: Row and column headings are coded with the th element

– Not applicable

Background

In order for users to correctly interpret the content of tables, it is important that tables have clear column and row headings. These headlines should not only be visually clear, they must also be structurally correct with the th element instead of td elements. Th indicates that it is a table header cell, as opposed to the td element which indicates that the cell is a table data cell.

All table headings must be coded as th element.

```
<table>
```

```

<caption>Flight prices Stockholm - Oslo</caption>

<tr>
  <th>Morning flights</th>
  <th>Afternoon flights</th>
  <th>Evening flights</th>
</tr>

<tr>
  <td>1500:-</td>
  <td>1750:-</td>
  <td>1750:-</td>
</tr>
...
</table>

```

The image below shows an example of column headers.

Flight prices Stockholm - Oslo		
MORNING FLIGHTS	AFTERNOON FLIGHTS	EVENING FLIGHTS
1500:-	1750:-	1750:-
150€	175€	175€
150\$	175\$	175\$

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

TB70: The scope attribute is used to indicate the direction of row and column headings when needed

– Not applicable

Background

Table headings must be correctly linked with the data cells that are subordinate to the header. For simple one-axis tables, it is sufficient if the table header is identified as th. In tables with both row and column headers, this element must be supplemented with the scope attribute.

For column headings scope="col" should be used and for row headings scope="row" should be used.

Example table with row and column headings:

```
<table>
  <caption>Flight prices Stockholm - Oslo</caption>

  <tr>
    <td></td>
    <th scope="col">April</th>
    <th scope="col">May</th>
  </tr>

  <tr>
    <th scope="row">Morning flights</th>
    <td>1500:-</td>
    <td>1750:-</td>
  </tr>
  ...
</table>
```

The image below shows an example of column and row headings.

Flight prices Stockholm - Oslo

	APRIL	MAY
MORNING FLIGHTS	1500:-	1750:-
AFTERNOON FLIGHTS	1250:-	Sold out
EVENING FLIGHTS	No departure	1750:-

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

TB100: Complex tables are supplemented with the relevant code

– Not applicable

Background

Complex tables must have more specific links between headers and data cells than simple tables. This can be achieved by giving each header an unique ID, and then referring to it in each cell using the header attribute.

In complex tables, logical categories can also be created using the axis attribute, which is used to group headings into common categories.

Examples of a more complex table where header cells and data cells are linked together:

```
<table>
  <caption>Flight prices Stockholm - Oslo</caption>

  <tr>
    <td></td>
    <th colspan="2" id="airline1">Funka airlines</th>
    <th colspan="2" id="airline2">Fake airlines</th>
```



```

</tr>

<tr>
  <td></td>
  <th id="month1" headers="airline1">April</th>
  <th id="month2" headers="airline1">May</th>
  <th id="month3" headers="airline2">April</th>
  <th id="month4" headers="airline2">May</th>
</tr>

<tr>
  <th id="time1">Morning flights</th>
  <td headers="airline1 month1 time1">1500:-</td>
  <td headers="airline1 month2 time1">1750:-</td>
  <td headers="airline2 month3 time1">1250:-</td>
  <td headers="airline2 month4 time1">1750:-</td>
</tr>
...

</table>

```

The image below shows an example of an advanced table.

Flight prices Stockholm - Oslo

	FUNKA AIRLINES		FAKE AIRLINES	
	APRIL	MAY	APRIL	MAY
MORNING FLIGHTS	1500:-	1750:-	1250:-	1750:-
AFTERNOON FLIGHTS	1250:-	Sold out	1500:-	Sold out
EVENING FLIGHTS	No departure	1750:-	No departure	1750:-

If you have such a complex table, your first choice would be to divide it into several, simpler tables or to simplify the layout in another way so that you can manage with only scope to associate headings and data cells.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.3.1 Info and Relationships \(A\)](#)

[EN 301 549 - 9.1.3.1](#)

Script & WAI-ARIA

SW20: Use WAI-ARIA in order to convey information that cannot be conveyed through the use of HTML and that benefits users

– Not applicable

Background

The WAI-ARIA standard includes a set of attributes that can be used to clarify the function, value, and role of various HTML elements. Some of the functionality covered by WAI-ARIA already has well-established solutions in HTML5, and in such cases, the HTML solution should be used. However, much of WAI-ARIA enhances accessibility beyond what is natively built into HTML5.

One area where HTML5 and WAI-ARIA overlap is navigation. The primary approach should be to use the HTML5 <nav> element to indicate a navigation menu. However, WAI-ARIA also provides an alternative with the role="navigation" attribute. Our recommendation is to prioritize the HTML5 solution whenever available.

A clear example of how WAI-ARIA improves usability is the aria-expanded attribute. This is used on buttons that expand or collapse sections, such as a mobile menu button. By setting aria-expanded="true" when the menu is expanded and aria-expanded="false" when it is collapsed, you provide essential feedback to users who rely on assistive technologies, helping them understand whether the menu is open or closed.

Avoid overusing WAI-ARIA

While WAI-ARIA offers many useful features, it is important not to overuse the standard. Do not add attributes unless they provide a clear benefit to users. Avoid

using WAI-ARIA when HTML alone is sufficient to convey functionality. For example, writing `<button role="button">` is unnecessary and can lead to redundant screen reader announcements. Similarly, do not use `role="menu"` and `role="menuitem"` for standard website navigation menus. These roles are intended for application interfaces, such as text editors, and may cause confusion for users by making assistive technologies read out "menu item" for each link in a standard menu.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 4.1.2 Name, Role, Value \(A\)](#)

[EN 301 549 - 9.4.1.2](#)

SW50: When new elements are added to the page, they are placed in the right order, both structurally and visually

✓ No errors found

Background

Occasionally, new content or new functions are added to a page without it being reloaded. When this happens, it is important that the new function or information appears in the right place both visually and structurally. If search results suddenly appear in a new area on the screen when the user starts writing in the search field, it is important that this area is clearly positioned visually where it belongs, and also in the right place in the structure so that assistive technology structurally present the information in the right place. It is best if this is done so that the new code is injected in the right places, for example:

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

There are also other ways to resolve this, such as sensing via script what the user

is doing and switching the focus so that the effect is the same for the user. How this works in practice should also be tested with the most commonly used assistive technology.

One common problem is that a new area is positioned correctly visually, but ends up last on the page structurally. If a tool does not find something until after the page footer, there is a big risk that the user will not find it at all.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

[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 \(A\)](#)

[WCAG 2.2 - 1.3.2 Meaningful Sequence \(A\)](#)

[WCAG 2.2 - 2.4.3 Focus Order \(A\)](#)

[EN 301 549 - 9.1.3.2](#)

[EN 301 549 - 9.2.4.3](#)

SW60: When new content or areas are loaded, focus is moved to them immediately

✓ No errors found

Background

If a covering layer or a new area opens dynamically on the page, it is important that the user focus is immediately directed onto this (unless it is a disturbance to the user). Examples of exceptions are search suggestions. The user's focus should not automatically be switched to the search suggestions as soon as they appear as this would be interruptive.

In the case of users that navigate through the keyboard, it is important that they come directly to the new layer or content without having to tab to the bottom of the page. If this is a form, the focus can be set on the first field, otherwise, the first tab step should be after the link that opened the layer or resulted in the new area being read in, leading into the first link in the area.

Also bear in mind that the reading order must be correct for users with assistive technology. When they continue to read downwards on the page from the link that

opened the layer, reading should continue into the layer in question. You may need to test this with assistive technology.

If it relates to messages and dialogue boxes, WAI-ARIA roles for messages and message dialogues ought to be used. If it is a form, it ought to be the alertdialog role:

```
role="alertdialog"
```

Otherwise the alert role:

```
role="alert"
```

[Information about WAI-ARIA \(W3C website\)](#)

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 2.4.3 Focus Order \(A\)](#)

[EN 301 549 - 9.2.4.3](#)

SW70: When status is changed, users with assistive technologies are informed without focus being moved

× Fail

Background

Updates in the interface that take place without the page being reloaded are particularly difficult for users with severe visual impairment to perceive. It may be that there is a change in an ongoing process, such as an updated queue on a website for the sale of concert tickets. It can also be about informing the user about how many search results a search has resulted in or validating errors in a form. It is problematic for the user to quickly move to another part of the interface to be able to read a status message before it is outdated or has been updated.

Examples of status changes are:

- Information about the number of search results in a search
- A change occurs in a waiting application
- A change takes place in an ongoing process
- Errors occur in a form

When a status message is displayed on the screen, the user should be informed about it without having to move to another area on the page. Attributes role = "alert", aria-live, and aria-busy can be used to inform screenreader users of status messages displayed in other parts of the page than where the user currently is. This means that the user gets status messages, without having to search for changes or to update the page. Assistive devices for users with cognitive impairments can clarify, delay, or suppress status messages, depending on the user's preference.



Your search on **accessibility** resulted in 384 hits

→ Survey on procurement of **accessibility**

The International Association of **Accessibility** Professionals (IAAP) is performing a study to better define ... procurement specialist responsible for acquiring **accessible** ICT goods and services. A group of people working

```
<div class="medium-12 columns">
<h2 aria-live="assertive" aria-atomic="true">
Your search on
<strong><mark>accessibility</mark></strong>resulted in 384
hits
</h2>
</div>
```

Wai-aria attributes can be used in combination to make it clearer to users that a status update is particularly important. An example of this is to use the role="alert" and aria-live="assertive" attributes in messages that inform the user that they are being logged out.

The attribute aria-live has three modes that determine how high priority an update in an area should have:

- **Assertive** = An important message that must interrupt assistive technology as soon as possible.
- **Polite** = A less important message that should be presented as soon as the user pauses.
- **Off** = A message that does not need to be presented to the user.

This requirement does not apply to error messages in modal windows, content displayed or hidden by expandable/collapsible objects or content that requires user input. Thus, information about cookies does not count as a status message, because those require users to confirm their consent.

Comment

When logging in with an SMS code and clicking "Log in", we get to a new view without notifying the screen reader that anything has changed. The focus is suddenly moved to an unnamed input field. Visually, the screen displays a message indicating "Identification" and also that an SMS has been sent to your number, but this information is not conveyed to assistive technologies.

To improve accessibility, the visible text should also be the label to the input field. This would ensure that screen reader users receive the same important information as users who can see it.

13:17

100



Doro

now

Response by Doro Verification
Code: 14601356



Identification

A Text has been sent to your number
+46767619329

Log in(01:42)

Log in

From Messages
14601356

1

2

ABC

3

DEF

4

GHI

5

JKL

6

MNO

7

PQRS

8

TUV

9

WXYZ

0



We evaluate this criteria as failed.

To do

- Make sure that assistive technology users are also informed when something important happens in the interface.
- Also make sure that if you change the user's focus and put it on a form field, that form itself must have a clear name and purpose.

Links to guidelines

[WCAG 2.1 - 4.1.3 \(AA\)](#)

[WCAG 2.1 - 4.1.3 Status Messages \(AA\)](#)

[EN 301 549 - 9.4.1.3](#)

SW80: Content that is shown when an object receives focus is fully usable and reachable until the user shifts focus

✓ No errors found

Background

Users with magnifying tools only see a small part of the screen at a time, which makes it hard to overlook large areas. It's also problematic for users to find content that is outside the area currently enlarged. Users who navigate with a mouse or a keyboard and magnifying tools, that track mouse cursor and keyboard movement, must be able to move the focus without an area disappearing. This requirement applies both on hover with a mouse- and keyboard focus.

An example is mega menus, that is menus consisting of several levels and are displayed in large areas shown when the user hovers over the menu. They usually cover other content and disappear when the user's cursor moves outside the area. In order for users with magnifying tools to access all content, the user must be able to move focus freely within the area without closing it. The user should also be able to close the area in a simple way, for example with the escape key.

Areas not obscuring other content do not require a closing mechanism. An example is menus that, instead of being opened in a covering layer, are opened above the page's content which is moved down and placed below the menu. Also, error messages describing input errors need to have a closing mechanism.

This requirement does not apply to modal windows or tooltips added with the title attribute as these are implemented in the browser.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

[WCAG 2.2 - 1.4.13 Content on Hover or Focus \(AA\)](#)

[EN 301 549 - 9.1.4.13](#)

SW90: When a modal window is shown, focus stays in the window until it is closed

– Not applicable

Background

A common mistake when using modal windows is allowing users to move focus behind the modal itself. This creates issues for various groups of keyboard users. It is a requirement that keyboard focus remains within the modal until the user actively chooses to close it.

When a modal window opens, keyboard focus should be moved inside the modal. Depending on the design of the modal, focus should be placed at the top of the window.

Focus must then remain inside the modal for as long as it is open. Funkas' recommendation is to implement a "focus trap" using a script, but other solutions exist, such as setting `tabindex="-1"` on all focusable elements outside the modal.

When the user closes the modal, focus should return to the element that originally opened it. Focus should not disappear entirely. We also recommend that the close button includes functionality to allow closing the modal with the Escape key.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 2.4.3 Focus Order \(A\)](#)

[EN 301 549 - 9.2.4.3](#)

SW100: Use aria-expanded for areas that can be expanded and collapsed



Background

The aria-expanded attribute should be used for areas that can be expanded and collapsed. The attribute is set on the element that unfolds the area in question. Aria-expanded="false" indicates that the area is collapsed and aria-expanded="true" that it is expanded. For example, this could apply to an FAQ list where the answers are expanded when the user clicks on the question, or on a menu that expands into submenus.

This can apply to an FAQ list where the answers are expanded when the user clicks on the question or a menu that dynamically expands the next level.

Aria-expanded should only be used if this is done dynamically, i.e. not with a reloading of the page.

If the collapsible area is not directly adjacent to the element that uses aria-expanded, a link between them may be needed. The link should then be created with the attributes aria-controls or aria-owns.

Example collapsed area:

```
<div class="faq-question">
  <button class="toggle" aria-expanded="false">
    When should I use aria-expanded?
  </button>
</div>

<div class="faq-answer" style="display:none">
  <p>
    When an area can be expanded dynamically without
    page reloading.
  </p>
</div>
```

Example expanded:

```
<div class="faq-question">
  <button class="toggle" aria-expanded="true">
```

```
        When should I use aria-expanded?
    </button>
</div>

<div class="faq-answer" style="">
    <p>
        When an area can be expanded dynamically without
        page reloading.
    </p>
</div>
```

Comment

We found only one button in the app that expands additional information, the "Menu" button (see example below). This type of button should clearly convey its purpose, indicating that it expands a new section or area.

Currently, the button uses a gear icon and has the programmatically determined name "Item," which is incorrect and unhelpful for screen reader users. We recommend replacing the icon with a more commonly recognized menu icon and renaming it to "Menu" or "Navigation". Additionally, the button should include a state indicator, such as "collapsed."

With these changes, a screen reader would announce something like "Menu, button, collapsed" providing users with clear and accessible context.

18:48



Overview



Funka Testsson

+46767619313

Doro 8200

updated 4 seconds ago



98%

Battery



Stockholm

Location



Location accuracy: Precise



Nybrokajen 7

SE-111 48 Stockholm +46 8555 770 60

contact@funka.com

www.funka.com/en

Item, Batter



We evaluate this criteria as failed.

To do

- We recommend adding the state "collapsed" for the menu button. If the user still has focus on this button when the menu is open, it should change state to "expanded" or similar.
- We also recommend that you name the button "menu" or "navigation" but also that you think again about which icon you use here.

Links to guidelines

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

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

[WCAG 2.2 - 4.1.2 Name, Role, Value \(A\)](#)

[EN 301 549 - 9.4.1.2](#)

Automatic events

AE20: New windows and covering layers are not opened unless the user has chosen to open them

✓ No errors found

Background

New windows and layers that partly cover the web page itself create problems for several groups of users. They create confusion and are often difficult to close.

Users should therefore always be told that a new window or covering layer is going to be opened. Users should also be able to choose if a new window or layer should be opened.

This checkpoint is aimed at layers and windows that open automatically, for example, website surveys that open on top of the main window when the visitor opens a page or advert windows that are opened when the user accesses the website. Ordinary links that open new windows should announce this to the user. Generally speaking, new windows and covering layers should not pop up without the user actively clicking on a link that clearly states what is going to happen. However, an option is to be to let the user accept this type of occurrence in browser settings.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

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

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

[WCAG 2.2 - 3.2.1 On Focus \(A\)](#)

[WCAG 2.2 - 3.2.2 On Input \(A\)](#)

[WCAG 2.2 - 3.2.5 Change on Request \(AAA\)](#)

[EN 301 549 - 9.3.2.1](#)

[EN 301 549 - 9.3.2.2](#)

AE60: If automatic events or time limits are present, there is an option to pause or increase the time span

– Not applicable

Background

In the case of time limits, users should be offered at least one of the following alternatives:

- Disable the time limit
- Extend the time limit. Users can personally decide how long the time limit should last and the max value should be at least ten times longer than the default value
- Expand the limit. For instance, by displaying a warning a few minutes before the time limit expires, the user has an opportunity to extend this

Users can, for example, have the option of disabling, expanding, or extending the time limit via a button that is displayed next to the time limit. Another alternative is to display a message a minute or so before logging out where the user can choose to extend the login time.

The time limit should always be related to the latest user activity that the system can recognize, not to the entire time of log in. If, for instance, the user goes from step 1 to step 2 in service, this should set the count to log out back to zero.

If the nature of the time limit is such that the whole point of the feature would be nullified if it were postponed or changed, this can be an acceptable exception.

Similarly, if the time limit is 20 hours or longer.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

[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 \(AAA\)](#)

[WCAG 2.2 - 2.2.1 Timing Adjustable \(A\)](#)

[WCAG 2.2 - 2.2.2 Pause, Stop, Hide \(A\)](#)

[WCAG 2.2 - 2.2.4 Interruptions \(AAA\)](#)

[WCAG 2.2 - 3.2.5 Change on Request \(AAA\)](#)

[EN 301 549 - 9.2.2.1](#)

[EN 301 549 - 9.2.2.2](#)

AE80: If an area is updated without the page reloading, this should be indicated with WAI-ARIA

– Not applicable

Background

Aria-live states how important changes are in the interface. For example, it decides whether or not assistive technology should interrupt the user.

Use the following values:

- **Off:** Use for something that is updated frequently, for example, stock price information.
- **Polite:** Use if a change only needs to be notified if the user is not doing anything important.
- **Assertive:** Use when something important happens that the user should be notified about as soon as possible.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 4.1.2 Name, Role, Value \(A\)](#)

[EN 301 549 - 9.4.1.2](#)

Frames

FR20: Every frame's purpose is described with the title attribute

– Not applicable

Background

The purpose of each frame should be described in the title attribute. This information is presented by assistive technology for visually impaired users, and helps them understand whether they should go into the frame and read that web page, or continue on the current web page outside the frame.

```
<iframe
  title="News from Funka"
  src="funkanews.html">
</iframe>
```

If a video is shown in the frame, this should also be mentioned along with a brief description of the video's content.

```
<iframe
  title="Video about accessibility on Youtube"
  src="https://youtube.com/funka">
</iframe>
```

Frames whose only purpose is to collect statistics should be hidden both visually and for assistive devices.

```
<iframe
  title="Statistics"
  src="statistics.html"
  aria-hidden="true">
</iframe>
```

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 4.1.2 Name, Role, Value \(A\)](#)

[EN 301 549 - 9.4.1.2](#)

Search functions

SF05: There is a search function, unless there is a specific reason to not have one

– Not applicable

Background

To meet the requirements of WCAG at level AA, there should be several different ways to find a specific page in the interface. Exceptions are pages that are steps in an e-service or process.

If you do not offer a search function, you must offer some other alternative way to find a page on the website. This could be a sitemap or an A-Z function.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 2.4.5 Multiple Ways \(AA\)](#)

[EN 301 549 - 9.2.4.5](#)

Documents & other forms of media

DM10: Background noise can be easily turned off manually or turned off automatically within 3 seconds

– Not applicable

Background

Just as movements in the interface affect the visitor's ability to understand the information, background noise also affects the ability to concentrate.

For this reason, background sounds that starts automatically should stop after a maximum of 3 seconds. Alternatively, there must be a clear and easy method to stop the sound.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.4.2 Audio Control \(A\)](#)

[EN 301 549 - 9.1.4.2](#)

DM20: Audio-only or video-only has a nearby text description

– Not applicable

Background

Audio or video can be a good way to convey information. They provide the opportunity to explain complex relationships in a more understandable manner. However, there are instances where users may not be able or willing to access audio or video. This could be due to visual or hearing impairments, lack of equipment support, firewall restrictions, or poor connectivity.

Therefore, content in audio and video should always be clearly described. The level of detail in the description should be determined based on the importance of the content. The description can also include links to other parts of the interface where users can access corresponding information.

Audio

When information is presented through an audio clip, the equivalent information must also be available in text form. There should also be a clear connection between the audio clip and the alternative text. This could include a transcription of the audio.

Video

For prerecorded video content, there should be a text version of the content clearly linked in proximity. The text version doesn't need to be an exact word-for-word representation of the content, but it should provide equivalent information.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.2.1 Audio-only and Video-only \(Prerecorded\) \(A\)](#)

[EN 301 549 - 9.1.1.1](#)

[EN 301 549 - 9.1.2.1](#)

DM30: Captions reproduce audio information in the pre-recorded video and audio clips

– Not applicable

Background

If you have audio as part of a video or as an audio file, the content must be subtitled:

- For video, a strip of text synchronised with the audio is sufficient.
- In audio clips, it may be difficult to insert a text strip, but you can offer the user to read a text transcript of the audio information instead.
- If you have a music-only video, the name of the song or the fact that music is playing should be captioned to let the user know that nothing is being said.

What types of audio should be captioned?

The captioning should provide the information conveyed by audio. This means speech, but can also be other important sounds that contribute to the understanding of the content.

For example, the Swedish Transport Administration has an information film about supervised pedestrian crossings and how the sound signal is used by the severely visually impaired. If it is demonstrated via audio, it is important that the signals are also described via subtitles.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

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

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

[WCAG 2.2 - 1.2.2 Captions \(Prerecorded\) \(A\)](#)

[EN 301 549 - 9.1.2.2](#)

DM40: Information presented as audio in live broadcasts is transcribed with captions

– Not applicable

Background

Audio information in live broadcasts must also be captioned.

We recommend that you make a judgement as to whether it is reasonable to offer live subtitling in the case in question. However, to meet the requirements of WCAG at level AA, live broadcasts should be captioned.

NOTE: Certain types of content are exempt from the law on accessibility in digital public services, such as live broadcasts of time-dependent media.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.2.4 Captions \(Live\) \(A\)](#)

[EN 301 549 - 9.1.2.4](#)

DM100: Visual information in video is explained with audio

– Not applicable

Background

Video should be explained with audio, or audio description, so that a user who does not see the video is able to understand what is happening.

In many cases, this can be done by having a regular voiceover explaining what is happening, but in more complex films, a separate audio track may be needed for audio description.

Even in this case, you should make a judgement of reasonableness. But audio description is a requirement if you want to meet WCAG level AA.

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

[WCAG 2.2 - 1.2.3 Audio Description or Media Alternative \(Prerecorded\) \(A\)](#)

[WCAG 2.2 - 1.2.5 Audio Description \(Prerecorded\) \(AA\)](#)

[EN 301 549 - 9.1.2.3](#)

[EN 301 549 - 9.1.2.5](#)

Plain language & comprehensibility

US10: The user doesn't need to solve, remember, or translate any tasks to identify themselves

✓ No errors found

Background

Some users with cognitive disabilities may not be able to solve puzzles, memorize things, or transcribe texts. Therefore, there must be an alternative way to identify themselves.

The alternatives can be:

- Another identification method that does not rely on a cognitive function test.
- A mechanism to assist the user in completing the cognitive function test.
- The cognitive function test is to recognize objects.
- The cognitive function test is to identify non-text content that the user provided to the website.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

WCAG 2.2 - 3.3.8 Accessible Authentication (Minimum) (AA)

US20: Instructions are understandable regardless of the user's ability to see or hear

✓ No errors found

Background

Everyone should have the opportunity to understand and manage instructions.

For this reason, instructions must be understandable regardless of the user's ability to see or hear. Avoid formulations that only refer to colour, shape, direction, or sound, such as the "circle on the left."

Referring to a colour can sometimes be a good way to clarify something, but must be complemented by other information. Instead of "click the green button", the instruction could be "click the green button with the text 'Accept'". Then, in addition to the colour, there is also a text to help the user find the button.

Do not refer to objects that are "to the right" or similar. The reference cannot be understood by users who cannot see the interface. In addition, depending on the

width of the screen in responsive interfaces, items may be placed elsewhere. Most instructional films assume that users can see and hear. Such films must therefore always be accompanied by a text version.

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 1.3.3 Sensory Characteristics \(A\)](#)

[EN 301 549 - 9.1.3.3](#)

US190: The primary language of each Web page can be programmatically determined

✓ No errors found

Background

The primary language must be specified in the code on all pages. This makes it possible for assistive technology to determine the language in which the text is written and thus select the correct voice.

The primary language should be specified directly in the HTML element at the top of the page code.

In HTML5, the main language is specified with the lang attribute:

```
<html lang="en">
```

In XHTML, the primary language is stated with the attribute xml:lang="en".

However, you should also add the older way of writing for backward compatibility (exception XHTML 1.1). Example:

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

Comment

The interface fulfills the requirement in the checkpoint satisfactorily on the parts we checked. We assess the checkpoint as "No errors found".

Links to guidelines

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

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

[WCAG 2.2 - 3.1.1 Language of Page \(A\)](#)

[EN 301 549 - 9.3.1.1](#)

US200: Content in a language that differs from the primary language of the page is programmatically determined

– Not applicable

Background

When language changes are made to the content, they should be programmatically determined as such.

For example, this could be a quote in another language, or that the page provides information in one language but the menus and site structure are in another. Such changes should be marked in the code.

This can be done on most elements, such as span, li, div, a and p. Individual terms and words do not need to be labelled.

In HTML5, language is indicated by the lang attribute:

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

This indicates that the paragraph in question is written in English. The attribute can be used on all HTML elements in HTML5.

In XHTML, language is specified with the attribute XML: lang = "en". However, this should be supplemented with the older way of writing for backward compatibility (exception XHTML 1.1). For example like this:

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

Comment

The interface does not use the type of solution that the requirement applies to. We assess the checkpoint as not applicable.

Links to guidelines

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

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

The Web Accessibility Directive and European Accessibility Act

WAD10: The Accessibility statement is accessible and easy to find

! No errors found but can be improved

Background

The Web Accessibility Directive requires all websites covered by the law to publish an accessibility statement. The accessibility statement should be easy to find on the home page or preferably accessible from all pages, for example in a footer. The link to this page should be named "Accessibility Statement."

The statement must be published in an accessible format and primarily target and support the end users of the website. It is also intended for the authority in charge of supervision.

Accessibility statement for apps

For a mobile application, the accessibility report must be published at least on the website of the responsible public actor, or where users can download the app.

The statement can also be in the app, but it is not sufficient if it this is the only available location.

Content of the accessibility statement

The report must contain

- the name of the authority
- the name of the national law
- the name of the app/digital service/website
- compliance with legal requirements
- list of what is **not** compliant and
- description of and link to the feedback function
- link to the complaint function of the supervisory authority
- date of compliance assessment and the method used
- date of publication and updates of the statement
- the methodology used to create the statement.

Recommendations (not requirements) to include in the accessibility statement

- link to the audit report.
- description of how the organization works with accessibility
- formal approval of the statement from the responsible person in the organization
- date when the website was published
- date of last significant update or redesign
- specific communication service for people with disabilities.

Comment

The requirement has the same assessment on iOS as on Android.

We have not found an accessibility statement for the app, but since you are not covered by the WAD, it is not relevant. However, if it becomes a requirement in 2025 in the upcoming EAA, it will be a must. However, publishing an accessibility statement is a great way to show awareness and commitment to accessibility.

For an app, the accessibility statement must at least be published on the website or where users can download the app. It can also be in the app, but not only there.

To do

- We recommend that you create an accessibility statement for the app.

WAD20: The feedback function or a link to such is found in the accessibility statement

– Not applicable

Background

Users should be able to inform about any accessibility issues through a direct feedback to the person responsible for the website.

Design of the feedback function

Users should be able to report errors in a feedback function. So that the visitor can contact the website owner and:

- report accessibility issues
- request accessibility of content exempted under certain sections of the law.

Please describe how the feedback function is handled and provide information on your normal response times. You can also provide the contact details of the department or person responsible for the website and feedback management.

WAD30: A link to the supervisory authority's notification function can be found in the accessibility statement

– Not applicable

Background

In cases where the visitor has provided feedback to the website owner about lack of accessibility but has not received a response, there should be a possibility to register their complaint with the supervisory authority.

The accessibility statement should describe how complaints are handled and provide a link to the complaints function of the supervisory authority.

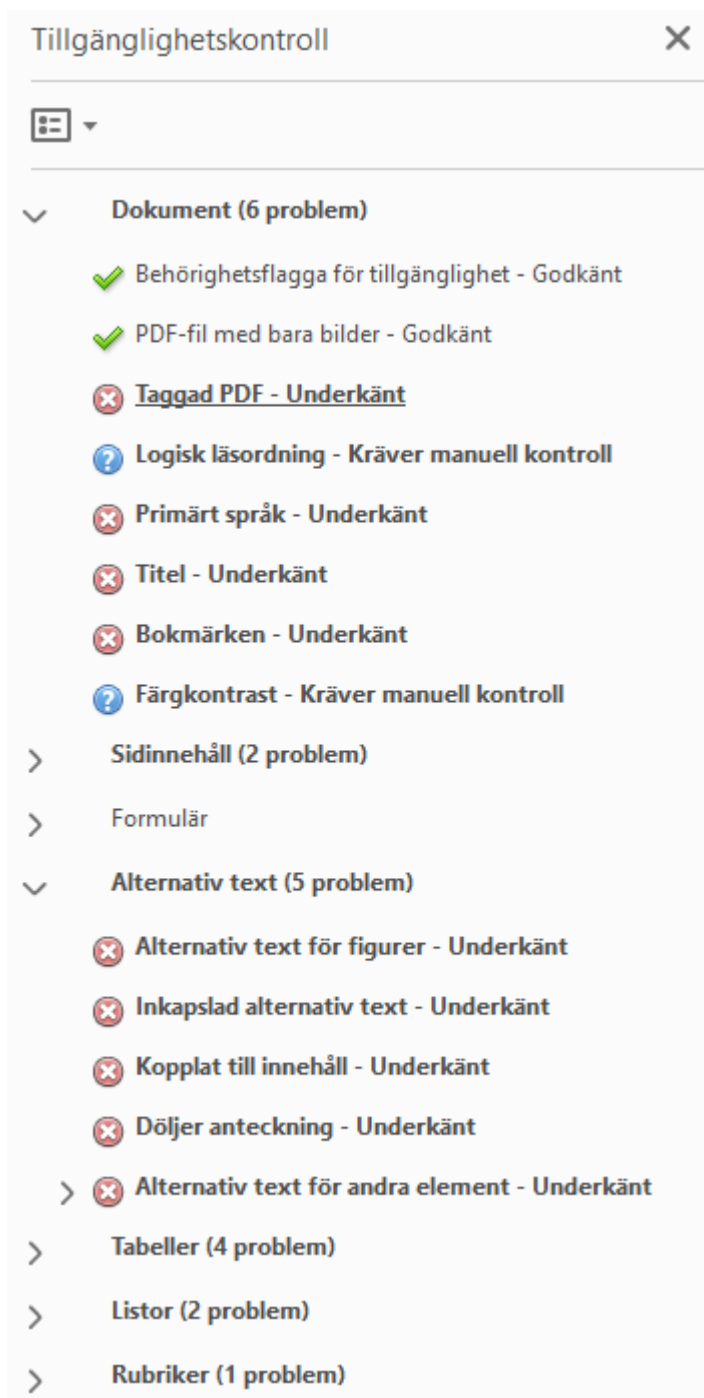
WAD50: Documents are accessible

– Not applicable

Background

Accessibility of various documents, such as PDF documents, is important and must not be forgotten. In this point we audit at least one document per service/main entrance and some randomly selected ones.

The image below shows an example of what an automatic accessibility check of a PDF document might look like when using the program Adobe Acrobat Pro.



You will benefit from requiring your PDF suppliers to ensure accessibility, and making sure your internal templates also create accessible PDF documents.

Comment

The interface does not use the type of documents that the requirement applies to. We assess the checkpoint as not applicable.