

Accelerating accessibility

in a **component-
based world**

Hydro de Vries • @hdv • SimpleWebConf • 24 June 2021

Hi, I'm Hidde.

Freelance accessibility and front-end consultant for organisations like the Dutch Government, Mozilla & W3C.

I write at hidde.blog.



What is accessibility?

Accessibility: to ensure people with disabilities can use your website.

Accessibility: to ensure people with disabilities can buy your products.

Accessibility: to ensure people with disabilities can complete all steps.

Accessibility: to ensure people with disabilities can use your service.

1 in 5

people on the planet
are disabled

43%

of your mobile users have
accessibility features turned on

@hdv

Data source: <https://toegankelijkheid.q42.nl/>



People with disabilities face barriers on the web. Even for simple tasks like transferring money and making a vaccin appointment.

We've GOT to smash those barriers!

Apple executives get 10% increase/decrease in bonus for performance around 'core values', including accessibility.



**Accessibility
standards**

+

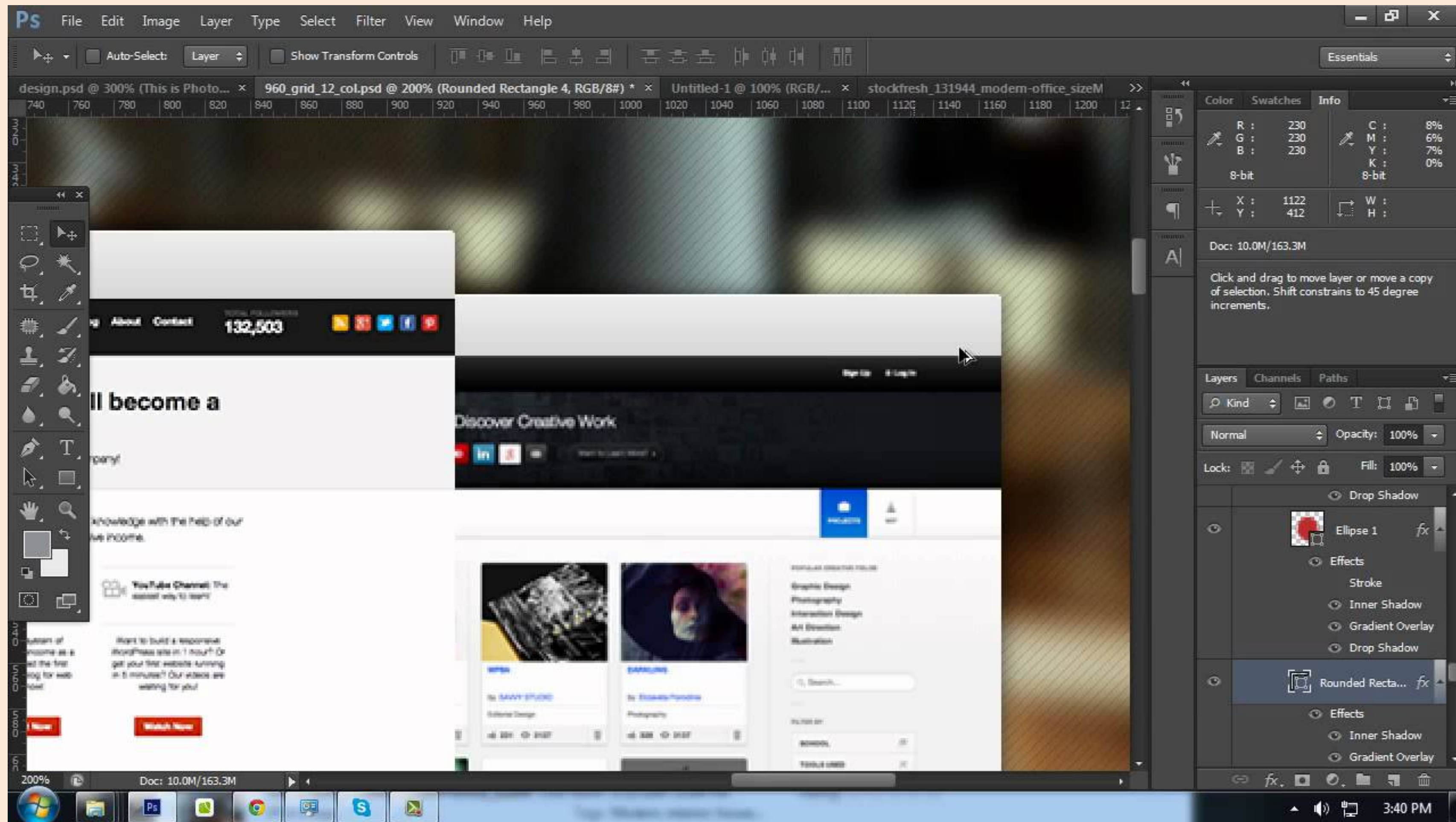
**Best
practices**

+

**User
testing**

(You'll need all of the above)

Let's talk components



Once upon
a time...

**Components changed
how we design, develop
and create for the web.**

@hdv

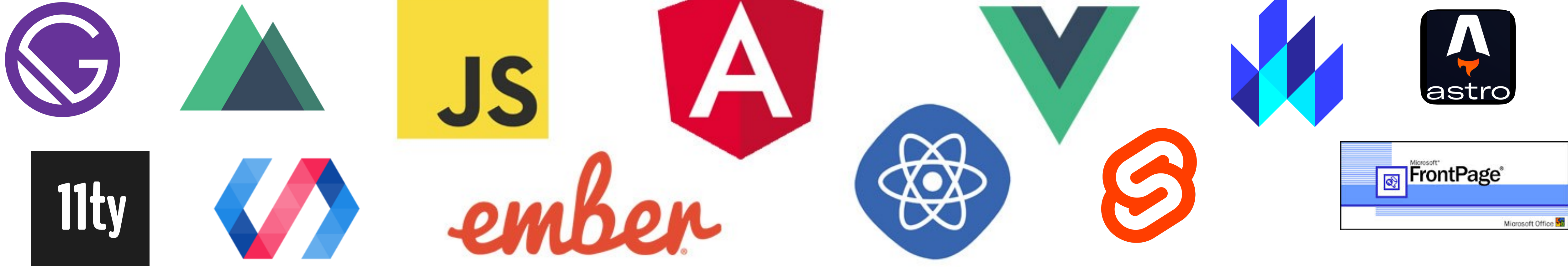


Reusability is key.

**With components, we can make
some accessibility reusable.**

don't repeat
inaccessible patterns

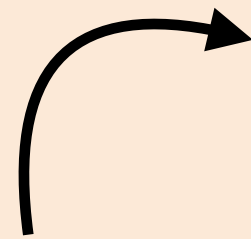
repeat
accessible patterns



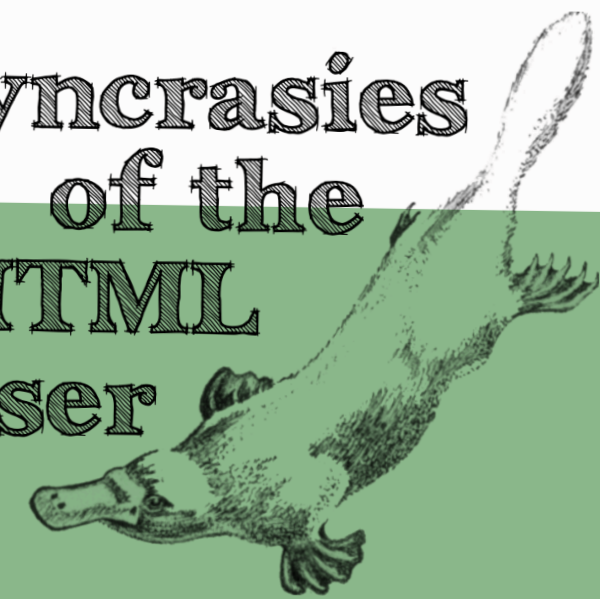
**Pick any framework.
It's the markup that matters.**

DOM tree

htmlparser.info



Idiosyncrasies of the HTML parser



Simon Pieters

@hdv

```
Elements Console Sources Network >> 6 | X X
[endif]-->
<!--[if IE 8]><html class="no-js lt-ie10 lt-ie9" lang="nl"><![endif]-->
<!--[if (gt IE 8) & (!IEMobile)]><html class="no-js lt-ie10 ie9-desktop"
lang="nl"><![endif]-->
<!--[if (gt IE 8) & (IEMobile)]><html class="no-js lt-ie10 ie9-mobile"
lang="nl"><![endif]-->
<!--[if !IE]><!-->
<html class="js applicationcache audio canvas hashchange history postmessage
no-touchevents video cssanimations backgroundsize bgsizcover borderradius
boxsizing csscalc csscolumns csscolumns-width csscolumns-span csscolumns-fill
csscolumns-gap csscolumns-rule csscolumns-rulecolor csscolumns-rulestyle
csscolumns-rulewidth csscolumns-breakbefore csscolumns-breakafter csscolumns-
breakinside flexbox flexboxlegacy fontface cssgradients rgba csstransforms
csstransitions localStorage" lang="nl">
  ▶ #shadow-root (open)
    <!--<![endif]-->
    ▶ <head>...</head>
    ▼ <body class="homepage cookiewall" data-comscore="{\"name\":
\"track.click.homepage\"}\" style="position: relative; margin-top: 474px;">
      ▶ <div id="npo_cc_notification" style="top: -474px;">...</div>
      ▶ <header id="nav" class="nav-wrapper">...</header>
      ▼ <main id="content" role="main">
        ::before
        ... ▼ <section id="topstories" class="js-topstories js-topstories-
interactive" data-comscore="{\"nos_origin\":\"topstory\"}"> == $0
          <h2 class="vh">Topstories</h2>
          ▶ <div class="topstories_wrapper topstories-twostories">...</div>
        </section>
        ▶ <div id="main">...</div>
        ▶ <section id="most_viewed_videos">...</section>
        ▶ <section id="editors_picks">...</section>
        ▶ <section id="nieuws_in_beeld">...</section>
        ▶ <section id="websites">...</section>
        ▶ <section id="categories">...</section>
        ::after
      </main>
html body main#content section#topstories.js-topstories.js-topstories-interactive
```

Accessibility Tree

The screenshot shows the Accessibility Inspector tool interface. At the top, there are icons for the Accessibility Inspector and Accessibility, along with a 'Check for issues: None' dropdown and a 'Simulate: None' dropdown. The main area displays the Accessibility Tree, which is a hierarchical structure of accessibility objects. The selected object is a link with the following properties:

- name: "Understanding Language of Page"
- role: "link"
- actions: [...]
- value: "<https://www.w3.org/WAI/WCAG22/Understanding/lang...>"
- DOMNode: a
- description: ""
- keyboardShortcut: ""
- childCount: 1
- indexInParent: 0

The tree structure above the selected link shows a 'section' containing the link, and the link itself has two 'text leaf' children with the text "Understanding Language of Page" and "" respectively.

Accessibility Tree

Names, roles, states

@hdv

The screenshot shows the Accessibility Inspector tool interface. At the top, there are tabs for 'Inspector' and 'Accessibility', with 'Accessibility' selected. Below the tabs, there are controls for 'Check for issues: None', 'Simulate: None', and 'beta Tabbing'. The main area displays the Accessibility Tree, which is a hierarchical structure of accessibility objects. The tree is expanded to show a 'link' object, which is highlighted in blue. The 'link' object has a 'text leaf' property with the value 'Understanding Language of Page'. Below the tree, there are sections for 'Checks' and 'Properties'. The 'Properties' section is expanded to show the following details for the selected 'link' object: 'name: "Understanding Language of Page"', 'role: "link"', 'actions: [...]', 'value: "https://www.w3.org/WAI/WCAG22/Understanding/lang..."', 'DOMNode: a', 'description: ""', 'keyboardShortcut: ""', 'childCount: 1', and 'indexInParent: 0'.

```
Accessibility Inspector
```

Check for issues: None | Simulate: None | beta Tabbing

Accessibility Tree

- section: ""
 - link: "Understanding Language of Page"
 - text leaf: "Understanding Language of Page"
 - text leaf: ""
 - text container: ""

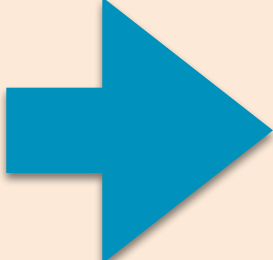
Checks

Properties

- name: "Understanding Language of Page"
- role: "link"
- actions: [...]
- value: "https://www.w3.org/WAI/WCAG22/Understanding/lang..."
- DOMNode: a
- description: ""
- keyboardShortcut: ""
- childCount: 1
- indexInParent: 0

Accessibility tree

```
Elements Console Sources Application >> 2 X
<ul id="menu" class="nav_list clearfix">
  ::before
  <li id="nav-nieuws" class="nav_item nav_item--hover js-menu-item">
    <a href="/nieuws/" class="nav_category js-event-click js-news js-dashboard-trigger" data-js-dashboard-target="nieuws" data-comscore="{\"nos_menu\":\"nieuws\"}>...</a> == $0
    <a href="#menu-nieuws" title class="nav_expand no-js-hidden js-submenu-toggle js-event-click" data-comscore="{\"nos_menu_nieuws\":\"mobile_toggle\"}>...</a>
    <div id="menu-nieuws" class="subnav nav_item_hover">...
  </li>
  <li id="nav-sport" class="nav_item nav_item--hover js-menu-item">...</li>
  <li id="nav-uitzendingen" class="nav_item nav_item--hover js-menu-item">...</li>
</ul>
#menu #nav-nieuws a.nav_category.js-event-click.js-news.js-dashboard-trigger
Styles Event Listeners DOM Breakpoints Properties Accessibility Accessibility Properties
Accessibility Tree
WebArea "NOS.nl - Nieuws, Sport en Evenementen | Nederlandse Omroep Stichting"
  Unknown
  banner
  navigation
  list
  listitem "Nieuws Overzicht □ Binnenland Buitenland Regio Politiek Economie Konin
  link "Nieuws"
  text "Nieuws"
ARIA Attributes
No ARIA attributes
Computed Properties
Name: "Nieuws"
aria-labelledby: Not specified
aria-label: Not specified
Contents: "Nieuws "
title: Not specified
Role: link
```

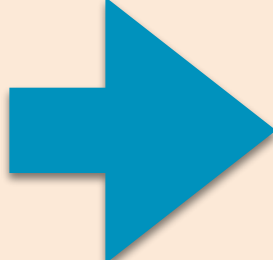


Platform APIs

 Microsoft
Microsoft Active Accessibility
Microsoft User Interface Automation
MSAA



Mac OS X Accessibility Protocol

 
Linux/Unix Accessibility Toolkit
IAccessible2



Assistive Technologies


JAWS for Windows

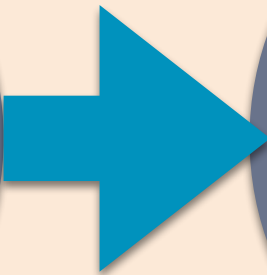
 

text-to-speech
screen magnifiers
alternate pointing devices

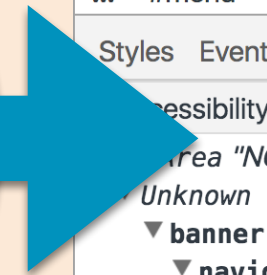
Accessibility tree

Platform APIs

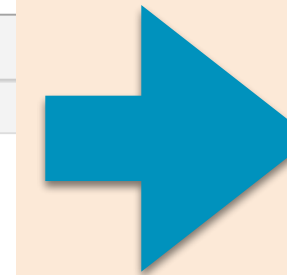
Your markup



DOM tree

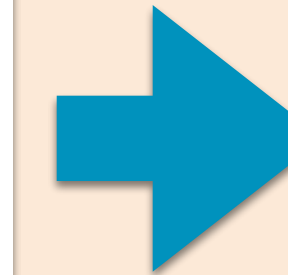


The screenshot shows a browser's developer tools interface. The top pane displays the DOM tree with a selected link element: `...`. The bottom pane shows the Accessibility tree for the same element, identifying it as a link with the text "Nieuws".



This panel lists platform-specific accessibility APIs:

- Microsoft Active Accessibility (MSAA)
- Microsoft User Interface Automation (MSAA)
- Mac OS X Accessibility Protocol
- Linux/Unix Accessibility Toolkit (IAccessible2)



The final step shows a partial view of an accessibility icon (a person in a wheelchair) and the word "alte".

**Some checks for
each component**

An accessible component...

**Works without
mouse**



Works without mouse: people

People who are blind

cannot use a mouse,
requires eye-hand
coordination

People with low vision

may have trouble
tracking pointer
indicator on screen

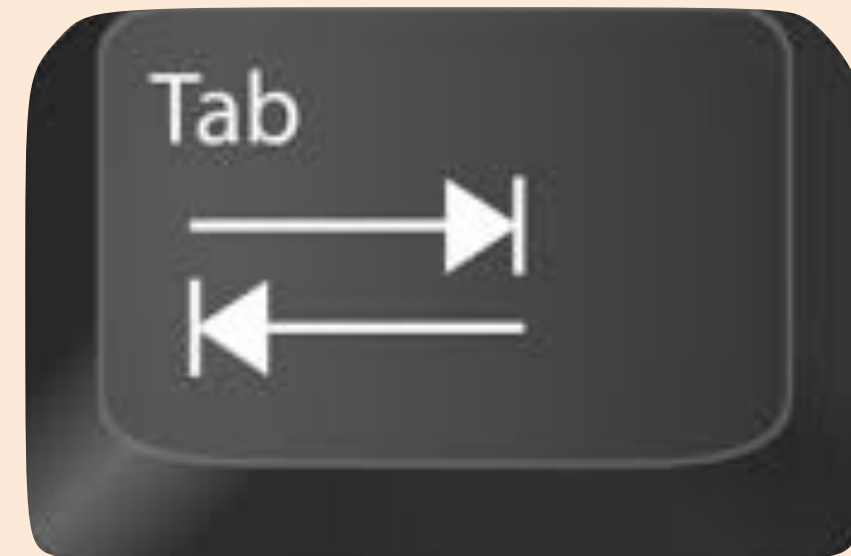
People with hand tremors

sometimes find using
a mouse very difficult,
prefer keyboard

Works without mouse: testing

Is it clickable?
It should also
be **TAB**-able

Links, buttons and
other controls



Works without mouse: testing

Is the active
control
highlighted?

Override the browser
default : focus
and make it shine

Find out more about what I do, my
background and experience or my
blog posts. Or contact me for
availability.

Works without mouse: testing

**Does the
order make
sense?**

Ensure a "logical,
usable source order"

An accessible component...

Has sufficient contrast

contrast

contrast

contrast

contrast

contrast

contrast

contrast

contrast

contrast

contrast

contrast

Has sufficient contrast: people

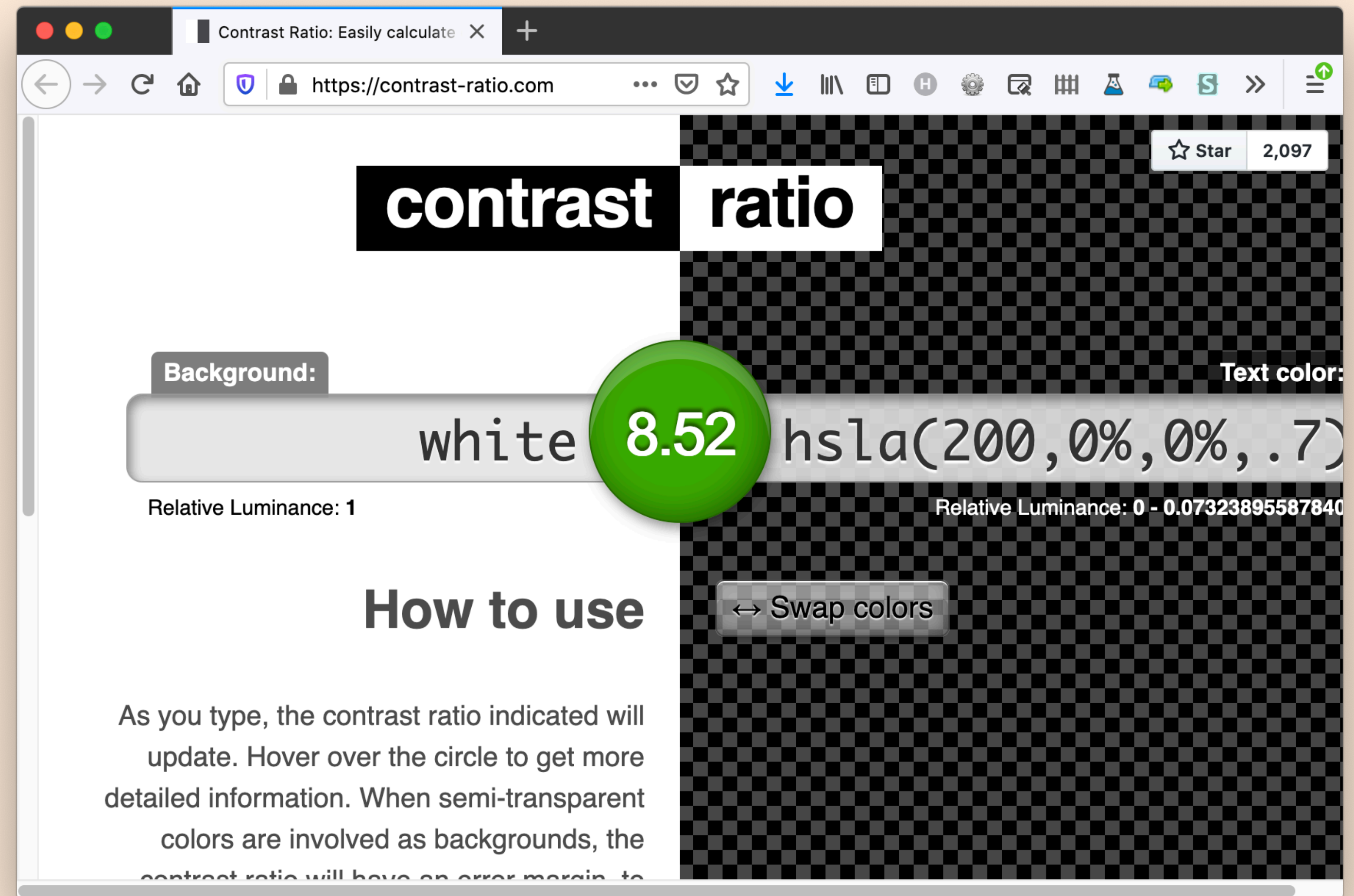
People with low vision or colour blindness

often find it difficult
to read text with low
contrast

Has sufficient contrast: testing

Use an automated contrast checker

like contrast-ratio.com or Firefox Developer Tools



Has sufficient contrast: testing

whocanuse

information about how well your colours work with common vision types

The screenshot shows a web browser window with the URL <https://whocanuse.com>. The page title is "Who can use this color combination?". The main results are displayed in a light blue box: "CONTRAST RATIO" is 8.41:1 and "WCAG GRADING" is AAA. Below this, a table lists various vision types with their respective population percentages and simulation buttons.

~POPULATION	VISION TYPE	SIMULATION
68%	Regular Vision (Trichromatic) AAA Can distinguish all three primary color, little to no blurriness	Text
1.3%	Protanomaly AAA Trouble distinguishing reds	Text
1.5%	Protanopia AAA Red blind - Can't see reds at all	Text
5.3%	Deuteranomaly AAA Trouble distinguishing greens	Text
1.2%	Deuteranopia AAA Green blind - Can't see greens at all	Text
0.02%	Tritanomaly AAA Trouble distinguishing blues	Text

An accessible component...

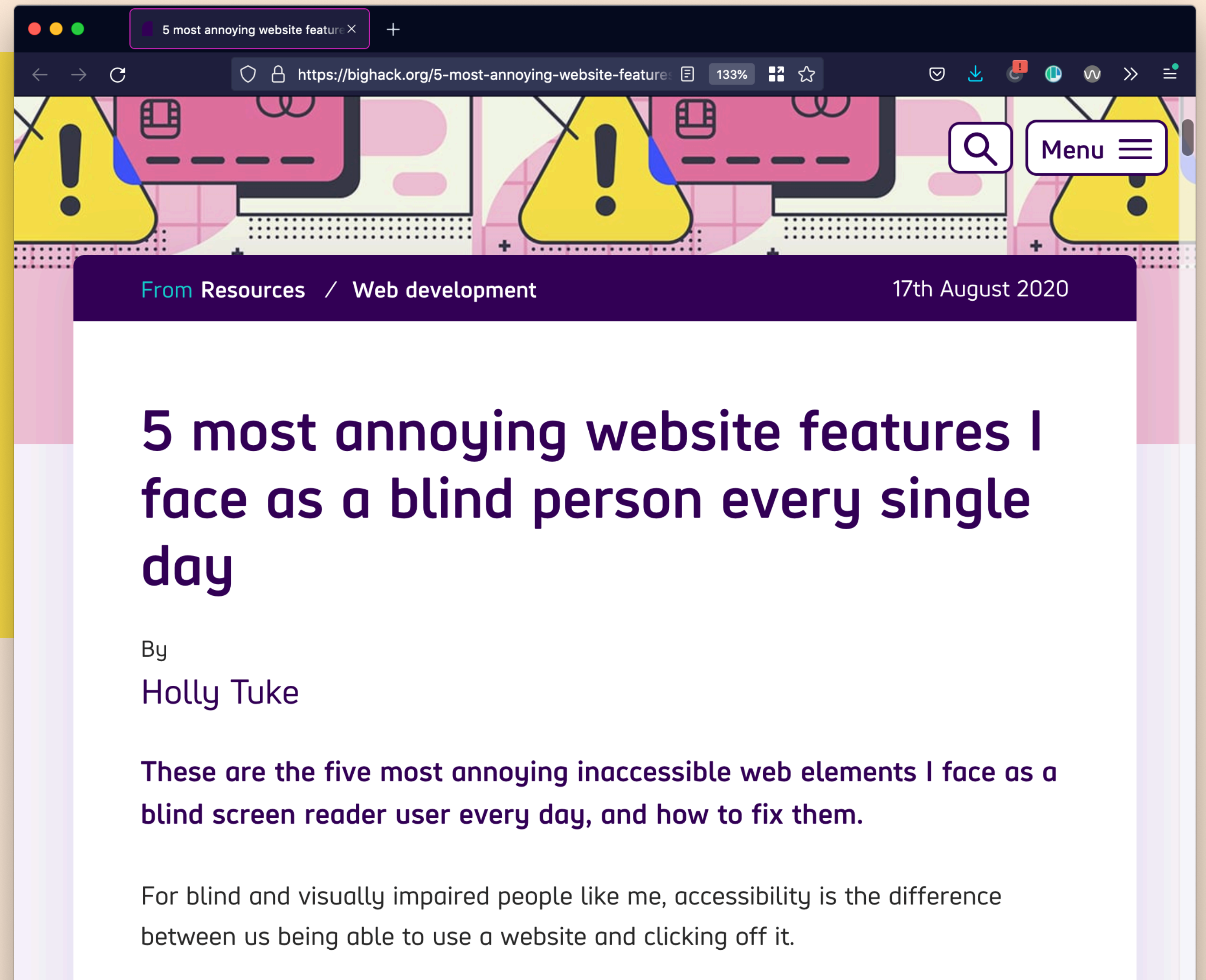
Names all controls

Names all controls: people



“unlabelled links make it much harder to navigate the website easily, quickly and independently”

– Holly Tuke, Life of a Blind Girl



Names all controls: people

People with physical disabilities

may use voice recognition software to interact

People who are blind

and use screenreaders

Names all controls: how to

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left

Names all controls: how to

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left

Names all controls: how to

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left

```
<button></button>
```

Names all controls: how to

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left

```
<button></button>
```

```
<!--
```

```
  Role: button
```

```
  Accessible name: null
```

```
-->
```


Names all controls: how to (text content)

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left

Send!1

```
<button>Send!1</button>
```

```
<!--
```

```
  Role: button
```

```
  Accessible name: Send!1
```


```
-->
```

Names all controls: how to (text + image alt)

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left

Send!1 

```
<button>Send!1  
  <img alt="airplane" />  
</button>
```

```
<!--
```

```
  Role: button
```

```
  Accessible name:
```

```
    Send!1 airplane
```

```
-->
```

Names all controls: how to (image alt)

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left



```
<button>  
  <img alt="airplane" />  
</button>
```

```
<!--  
  Role: button  
  Accessible name: airplane  
-->
```

Names all controls: how to (ARIA, option 1)

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left



```
<button
  aria-label="Send it!"
>
  <img alt="airplane" />
</button>

<!--
  Role: button
  Accessible name: Send it!
-->
```

Names all controls: how to (ARIA, option 2)

The Very Social Network

Tell me when you first used the Web without giving a date.

320 characters left



```
<button
  aria-labelledby="h"
>
  <img alt="airplane" />
</button>
<span id="h">Send!</span>

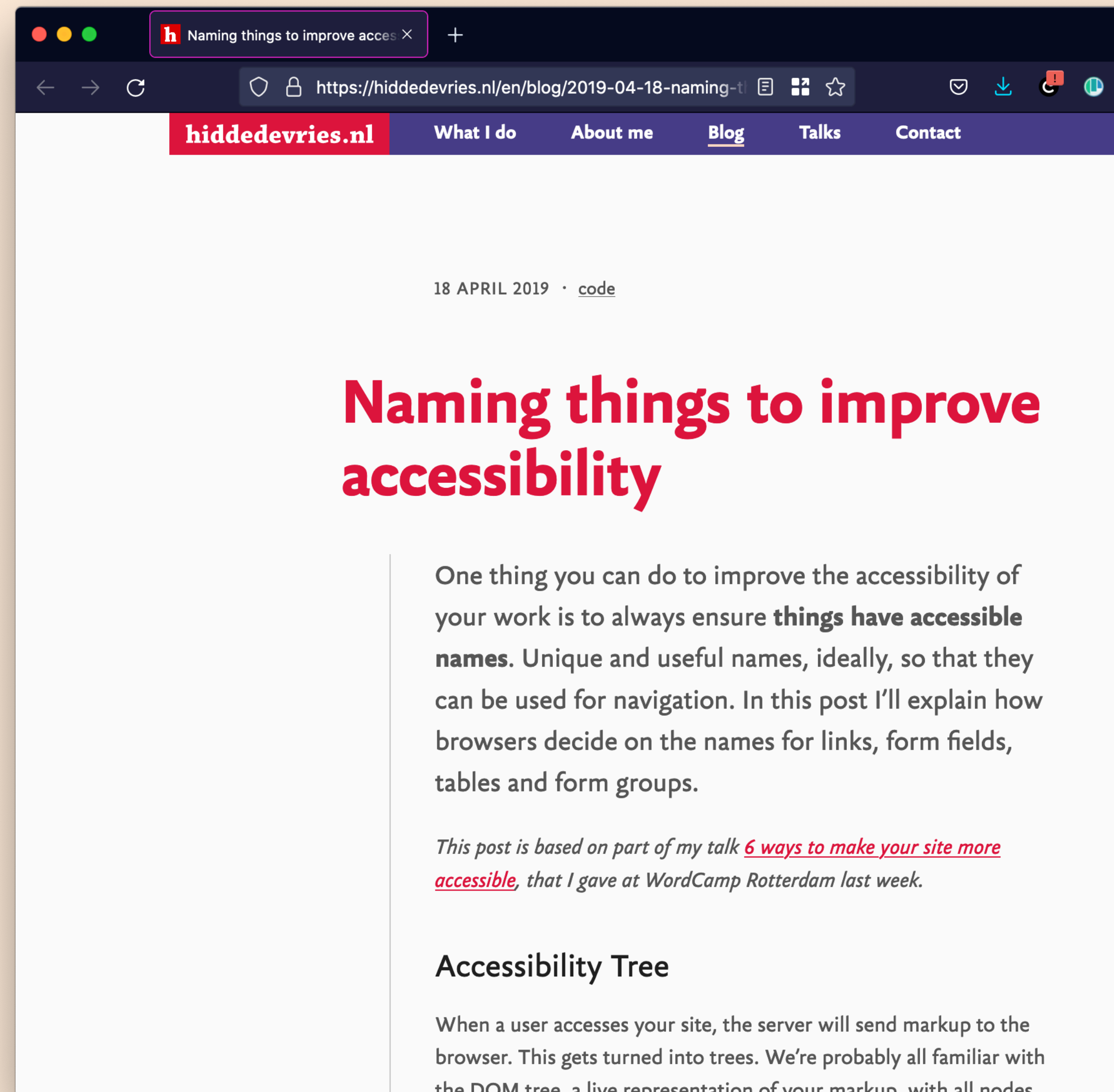
<!--
  Role: button
  Accessible name: Send!
-->
```

Names all controls

You'll want to ensure your component has useful names for all buttons and links.

For form fields, use `<label>`.

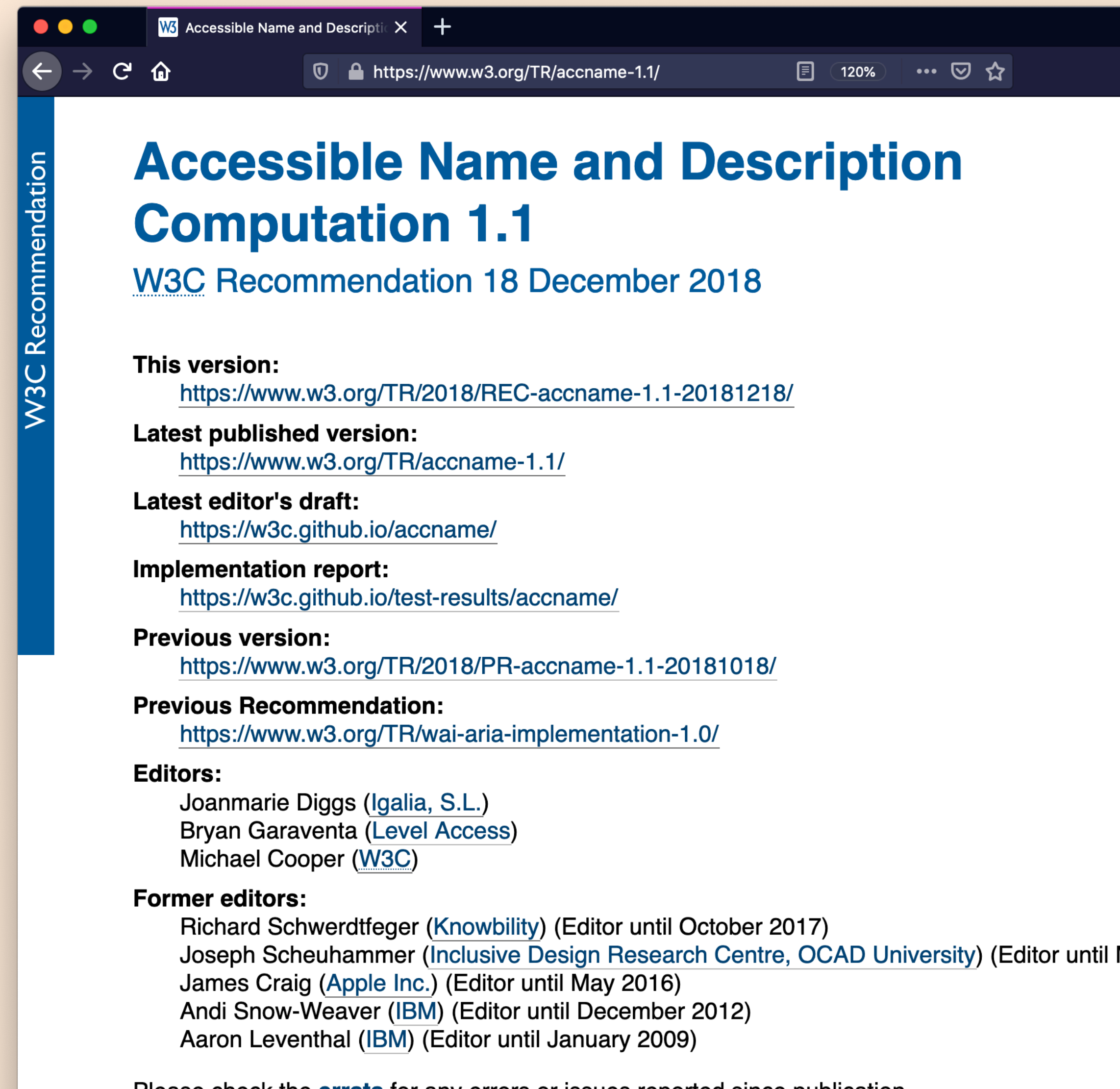
@hdv



Names all controls: how it is picked

1. Text content in control including `::before/::after` and alt text
2. `aria-label`
3. `aria-labelledby`

See: <https://www.w3.org/TR/accname-1.1/>



The screenshot shows a web browser window displaying the W3C Recommendation page for "Accessible Name and Description Computation 1.1". The page title is "Accessible Name and Description Computation 1.1" and it is dated "18 December 2018". The page includes links for the current version, latest published version, latest editor's draft, implementation report, previous version, and previous recommendation. It also lists the editors: Joanmarie Diggs (Igalia, S.L.), Bryan Garaventa (Level Access), and Michael Cooper (W3C). Former editors listed include Richard Schwerdtfeger (Knowbility), Joseph Scheuhammer (Inclusive Design Research Centre, OCAD University), James Craig (Apple Inc.), Andi Snow-Weaver (IBM), and Aaron Leventhal (IBM).

W3C Recommendation

Accessible Name and Description Computation 1.1

W3C Recommendation 18 December 2018

This version:
<https://www.w3.org/TR/2018/REC-accname-1.1-20181218/>

Latest published version:
<https://www.w3.org/TR/accname-1.1/>

Latest editor's draft:
<https://w3c.github.io/accname/>

Implementation report:
<https://w3c.github.io/test-results/accname/>

Previous version:
<https://www.w3.org/TR/2018/PR-accname-1.1-20181018/>

Previous Recommendation:
<https://www.w3.org/TR/wai-aria-implementation-1.0/>

Editors:
Joanmarie Diggs ([Igalia, S.L.](#))
Bryan Garaventa ([Level Access](#))
Michael Cooper ([W3C](#))

Former editors:
Richard Schwerdtfeger ([Knowbility](#)) (Editor until October 2017)
Joseph Scheuhammer ([Inclusive Design Research Centre, OCAD University](#)) (Editor until)
James Craig ([Apple Inc.](#)) (Editor until May 2016)
Andi Snow-Weaver ([IBM](#)) (Editor until December 2012)
Aaron Leventhal ([IBM](#)) (Editor until January 2009)

Please check the [errors](#) for any errors or issues reported since publication.

Names all controls: testing

**Is there a name
and what is it?**

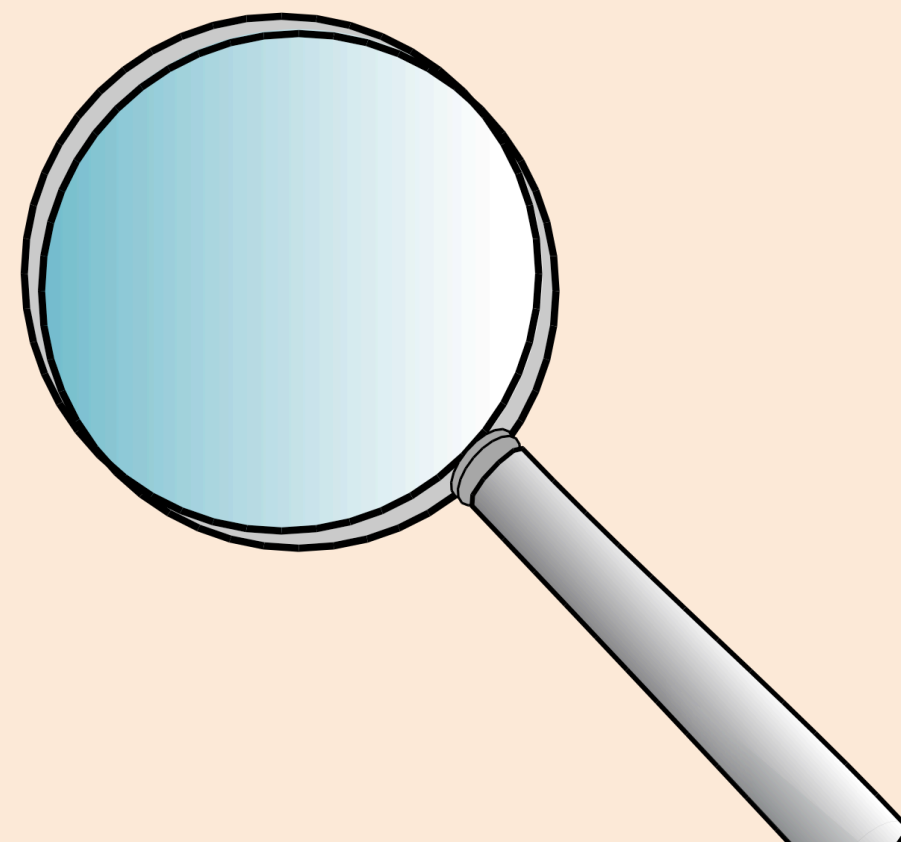
Check the accessibility
tree in the browser

**Is the name
meaningful?**

Function not form,
concise, no roles

An accessible component...

**Allows for
zoom**



Allows for zoom: people

People with low vision

who use zoom so
that they can read
the content

Allows for zoom: testing

**Use browser
zoom to 400%**

and verify nothing about
your components breaks



Previous image
Next image

An accessible component...

**Conveys states
to assistive tech**

Conveys states to assistive tech: people

People who use assistive technologies

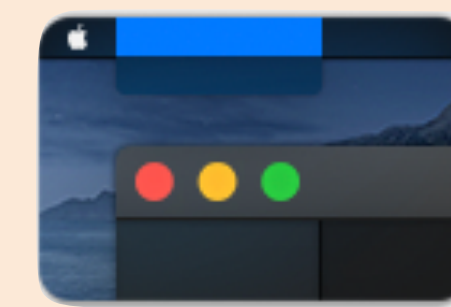
that are enabled by ARIA to
provide a much better UI

Conveys states to assistive tech: testing

Is it checked?	<code>aria-checked</code>
Is it disabled?	<code>aria-disabled</code>
Is it expanded?	<code>aria-expanded</code>
Is it hidden?	<code>aria-hidden</code>
Is it invalid?	<code>aria-invalid</code>
Is it pressed?	<code>aria-pressed</code>
Is it selected?	<code>aria-selected</code>

An accessible component...

Honours user preferences



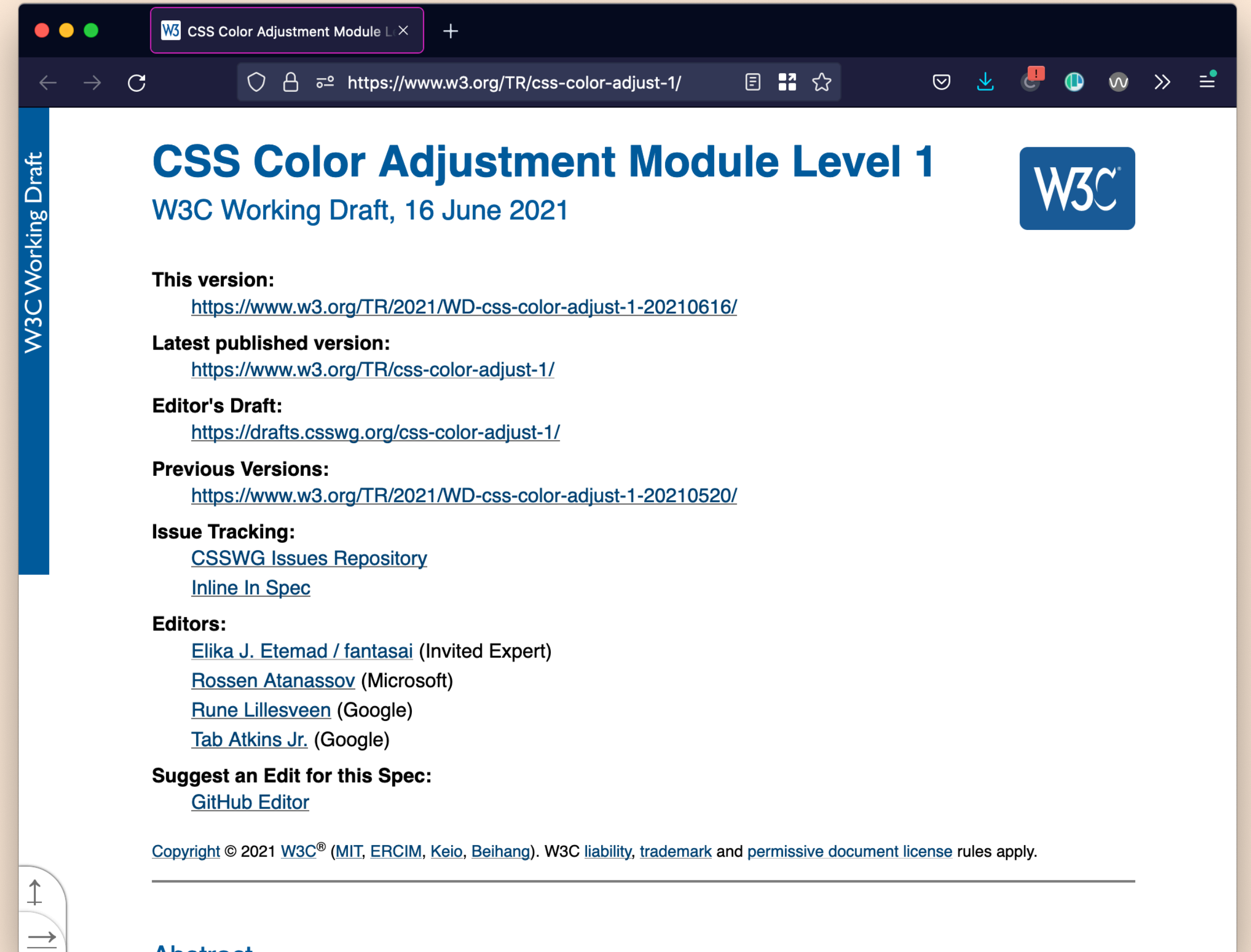
Honours user preferences: people

People with low vision

who use high
contrast and/or
forced color modes

Honours user preferences: background

Preferred color schemes
Forced color palettes



Honours user preferences: background

color, fill, stroke, text-decoration-color, text-emphasis-color, border-color, outline-color, column-rule-color, scrollbar-color, -webkit-tab-highlight-color, background-color, caret-color, flood-color, lighting-color, stop-color

Honours user preferences: background

**box-shadow, text-shadow, background-image,
color-scheme, scrollbar-color, accent-color**

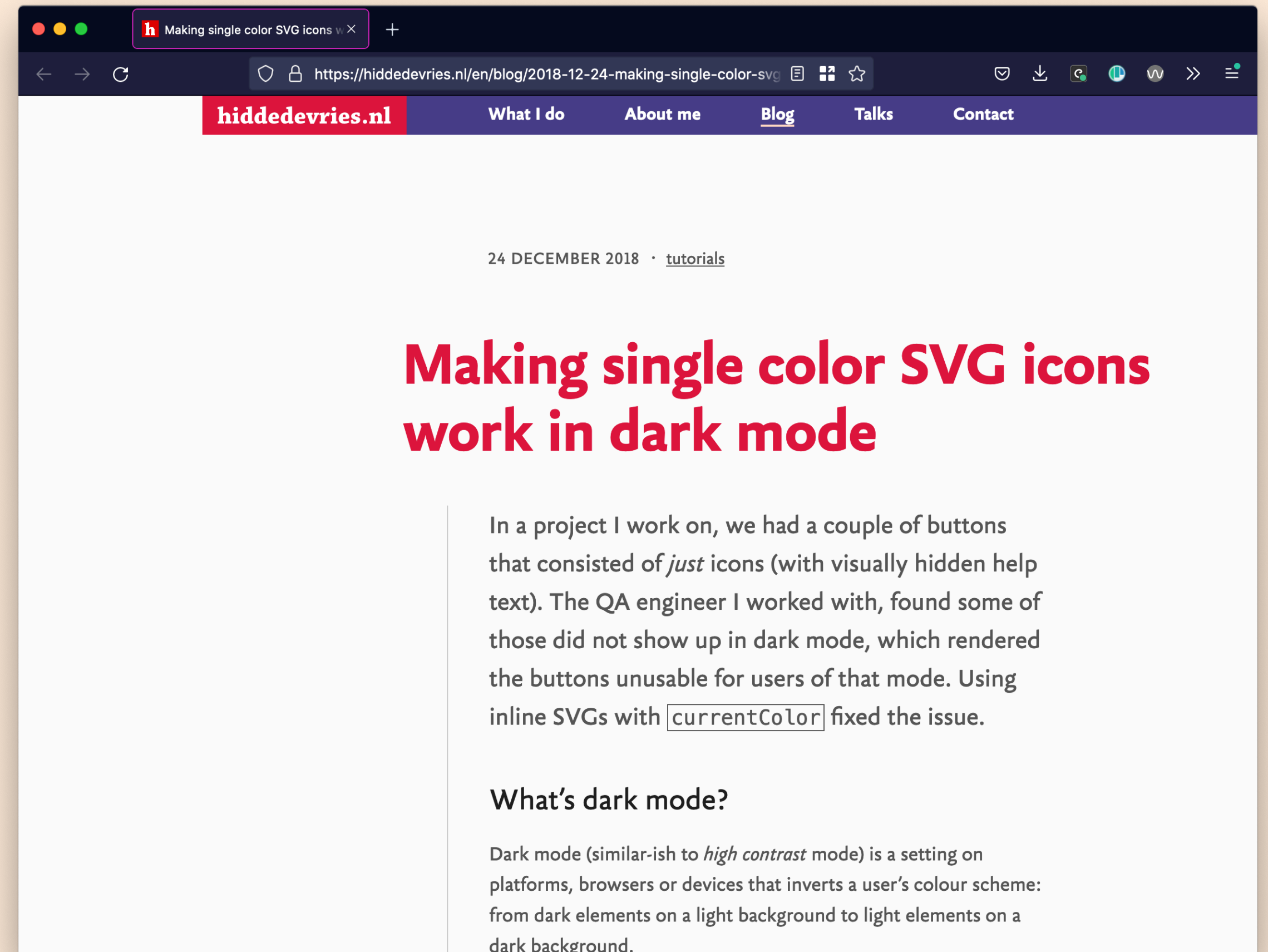
Honours user preferences: background

**box-shadow, text-shadow, background-image,
color-scheme, scrollbar-color, accent-color**

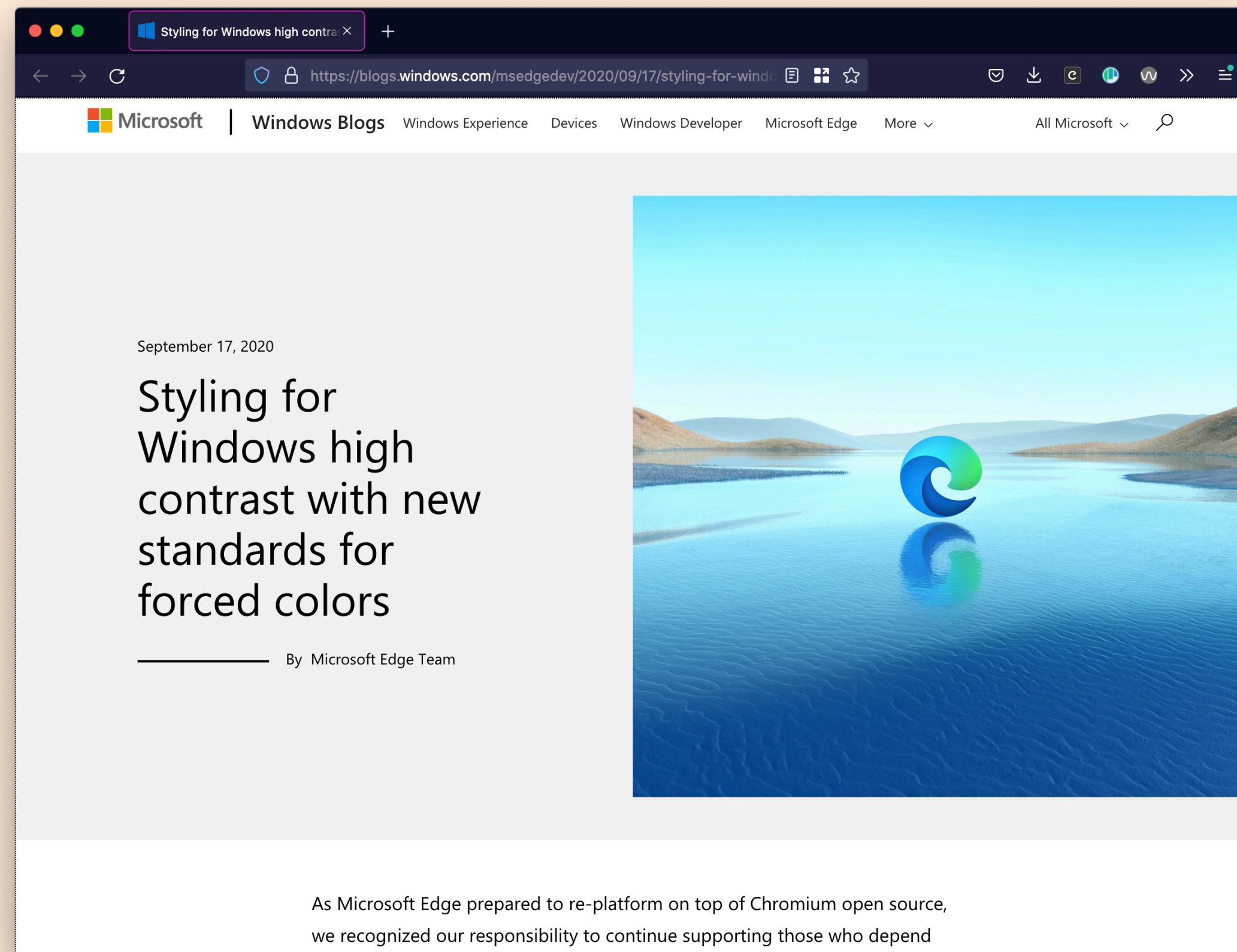
if you use this for a focus outline
and turn off `outline`, make sure
outline is `transparent` not `none`

Honours user preferences: how to

In SVG, use `currentColor` for fills and strokes



Honours user preferences: how to



“web developers can now use new web standards to style their content for forced color modes like Windows high contrast”

– Melanie Richards & Alison Maher



Accessibility Object Model

“

***“a JavaScript API to allow developers to modify
(and eventually explore) the accessibility tree for
an HTML page”***

– The AOM explainer document

“

“[AOM] fills the gaps in ARIA (...) is an API to provide your own accessibility (...) lets authors test them from JavaScript”

– Domenic Mizzone, Google

Setting semantics *without* mark-up

```
const e1 = document.querySelector("e1");  
  
e1.role = "button";  
e1.ariaDisabled = false;
```

Avoids "sprouting"

```
<my-custom-element  
  role="button"  
  aria-disabled="false"  
  ...  
>
```

Relationships without IDREFs

aria-activedescendant
aria-colcount
aria-colindex
aria-colspan
aria-controls
aria-describedby
aria-details
aria-errormessage

aria-flowto
aria-labelledby
aria-owns
aria-posinset
aria-rowcount
aria-rowindex
aria-rowspan
aria-setsize

Events from Assistive Technologies

Serious privacy concerns

```
// Implementing a canvas-based
// spreadsheet's semantics
canvas.attachAccessibleRoot();

let table = canvas.accessibleRoot
  .appendChild(new AccessibleNode());

table.role = "table";
table.colCount = 10;
table.rowCount = 100;

let headerRow = table.appendChild(
  appendChild(new AccessibleNode())
);
```

@hdv

Non-DOM nodes in the Accessibility Tree

**Will probably not happen, due to
complications including privacy**

Example from: AOM explainer • <https://wicg.github.io/aom/explainer.html#the-accessibility-object-model>

Reading accessibility tree through JavaScript

Google Docs Switches to Canvas X

https://thenewstack.io/go


THENEWSTACK Podcasts Events Ebooks Newsletter Sponsorship

Architecture Development Operations

DEVELOPMENT / TOOLS

Google Docs Switches to Canvas Rendering, Sidelining the DOM

24 May 2021 8:27am, by Richard MacManus



Earlier this month, Google [announced](#) it was updating the way Google Docs

**Warning: AOM
is not ready,
only for very
advanced uses**

<https://thenewstack.io/google-docs-switches-to-canvas-rendering-sidelining-the-dom/>

Summary

We've got to build **accessible** components

It's the **markup** that matters

Use **standards, best practices** and **user tests**

Consider **keyboard, contrast, names, zoom, states** and **user preferences**

THANKS!



SECTION LOOKING EAST