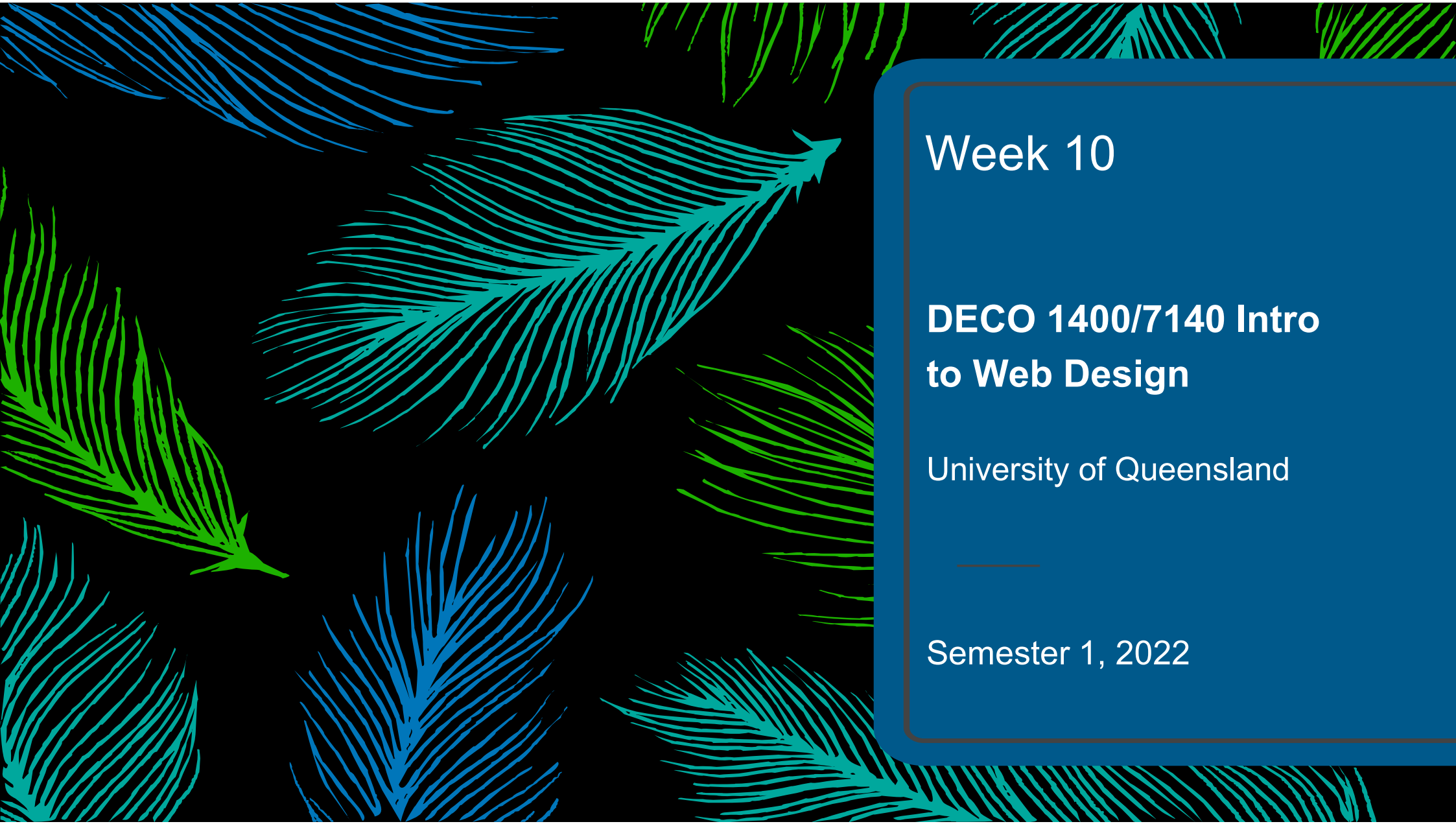


Week 10

DECO 1400/7140 Intro to Web Design

University of Queensland

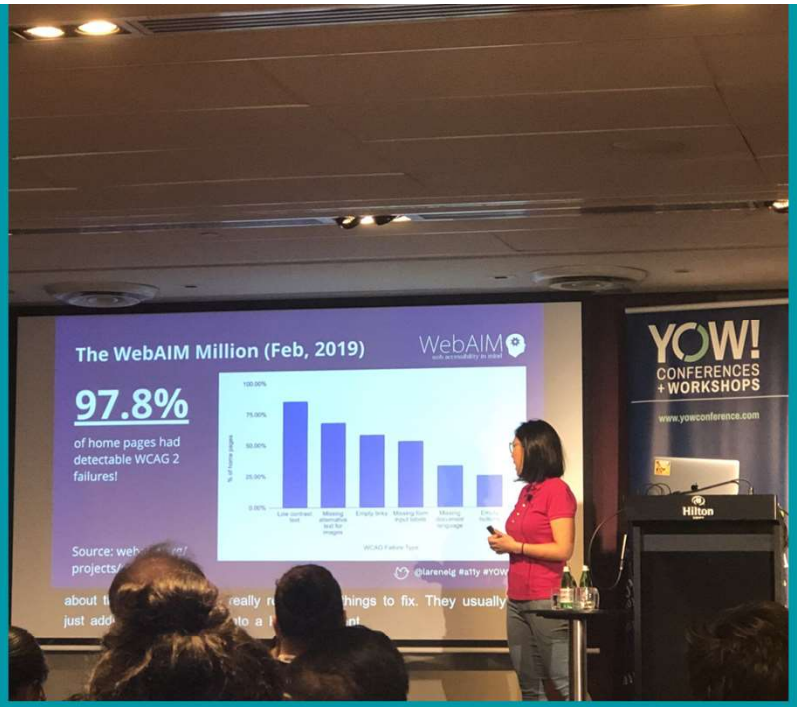
Semester 1, 2022


Guest Lecture on
Web Accessibility
(Sem 2, 2021)

Building an inclusive web

Larene Le Gassick
Software Engineer

@larenelg



About me

@larenelg

In June, I had a tweet go viral

@larenelg

@larenelg

Source: https://twitter.com/LareneLg/status/1270578058714443776

What is accessibility (#a11y)?

@larenelg

Unless software developers are deliberate about accessibility, we are going to exclude people.

@larenelg

Sadly, the state of web accessibility is not good.

70% of the web is inaccessible.

@larenelg

Source: Deque - The Internet is Unavailable (Aug, 2019)

For Global Accessibility Awareness Day (#GAAD) I authored a tweet series called "5 days to #a11y"

@larenelg

I hope this talk helps you to develop these good habits early.

@larenelg

@LareneLG #GAAD

5 days to #a11y

1. Unplug your mouse
2.
3.
4.
5.

Spend 30 minutes a day this week, to build these everyday inclusive developer habits into your workflow.

@larenelg

Why unplug your mouse?

- It helps make sure you include **keyboard-only** users
  - screenreader users (mostly low-vision or blind people)
  - people with permanent physical disabilities
  - people with temporary physical injuries or conditions
- Essential for some, but benefits everyone
  - e.g. complex forms are faster to fill out using just a keyboard

How to test

Spend some time trying to navigate what you're working on, or browsing, using **only the keyboard**.

Use only Tab, Arrow, Space, and Enter keys.

Things to check for:

- Do you know where you are?
  - Is there a focus outline on the button/input/anchor link you're on?
- Can you use Space and/or Enter key on buttons and links?
- Does keyboard focus move to a modal or menu when you open it? Does it move back to the main page when you close it?

@larenelg

Make sure there is always a focus outline

The most common and frustrating keyboard navigation problem is not knowing where you are on the page, and this CSS is the most common culprit:

```
:focus {
  outline: none;
}
```

Because browser default focus is "ugly". **Please don't do this.**

Check out http://gov.uk for the best-looking keyboard nav I've seen.

@larenelg

@LareneLG #GAAD

5 days to #a11y

1. Unplug your mouse
2. Add colour and contrast checkers to your tool belt
3.
4.
5.

Spend 30 minutes a day this week, to build these everyday inclusive developer habits into your workflow.

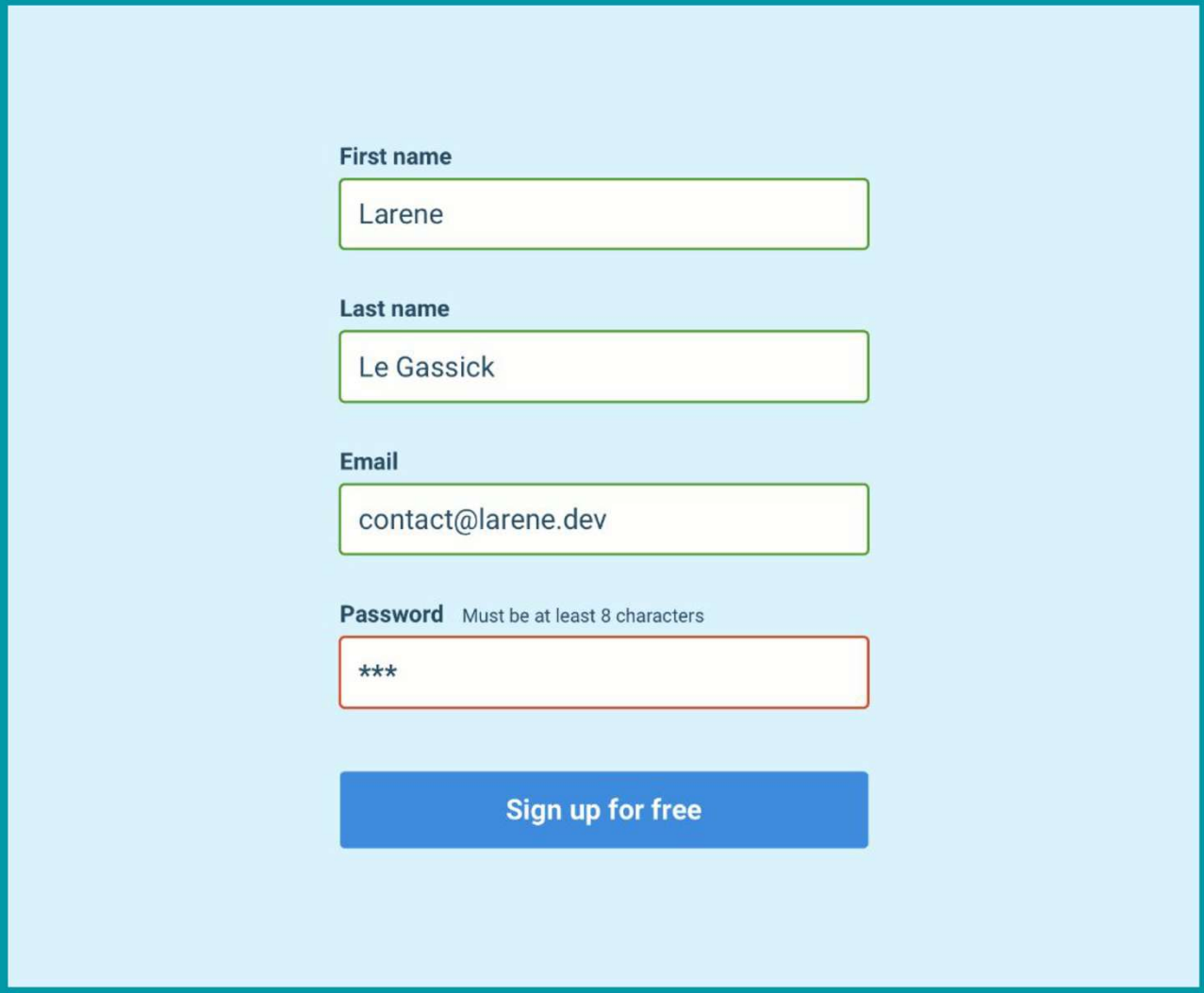
@larenelg

Colour

In Australia,
8% of men, and 0.4% of women have some degree of color blindness.

We cannot use only colour to provide meaning, or risk excluding 1-2 in every 20 people from the web.

@larenelg

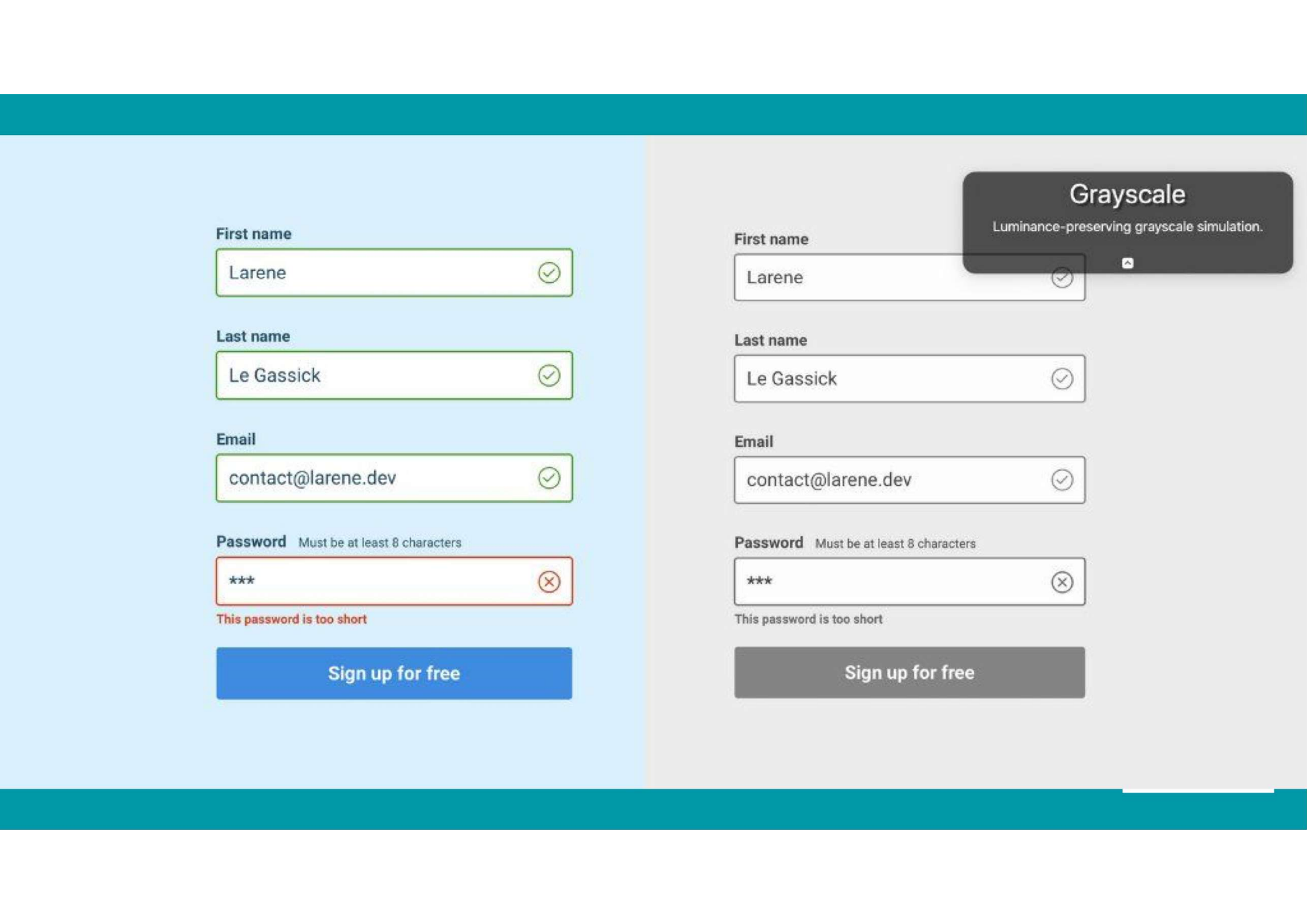

First name
Larene
Last name
Le Gassick
Email
contact@larene.dev
Password
Must be at least 8 characters
***
Sign up for free

@larenelg

Never rely on only colour for meaning.

Always use icons or text for meaning, then add colour.

@larenelg


First name
Larene
Last name
Le Gassick
Email
contact@larene.dev
Password Must be at least 8 characters
***
This password is too short
Sign up for free
Grayscale
Luminance-preserving grayscale simulation.
First name
Larene
Last name
Le Gassick
Email
contact@larene.dev
Password Must be at least 8 characters
***
This password is too short
Sign up for free

Contrast

Contrast

Contrast is the difference between luminance (light emitted) of one colour as compared to another.

For reference, the contrast ratio of

- white text on a white background is **1:1**
- black text on white is **21:1**

Web standards (i.e. WCAG) say

- accessible text/background contrast should be **at least 4.5:1**

Sadly, 86% of the web fail to meet minimum contrast ratio requirements as per to the Web Content Accessibility Guidelines (WCAG).

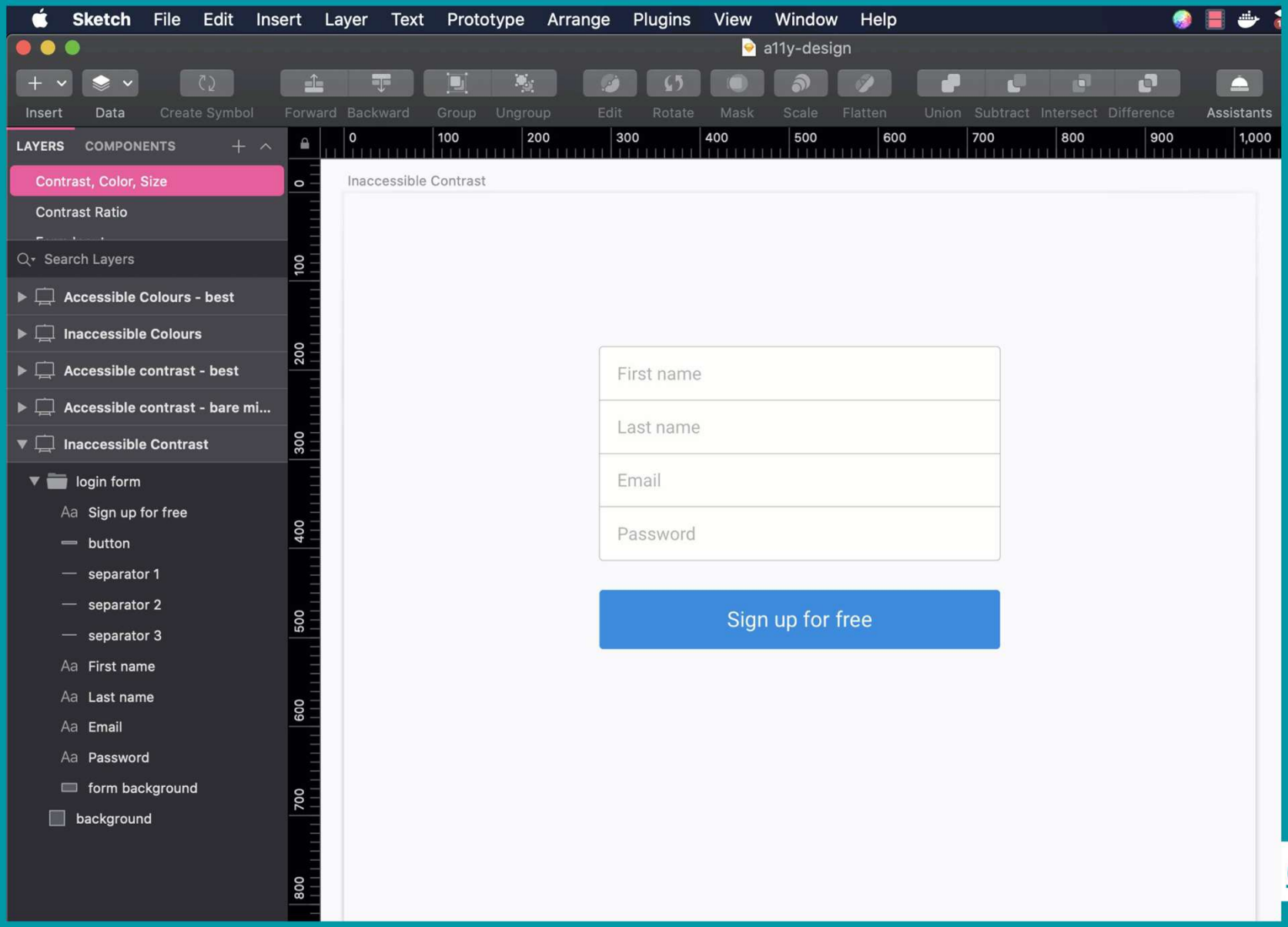
@larenelg

Source: WebAIM Million project

Good contrast

- is essential for people with ageing or low vision
- reduces eye strain
- is easier to see in harsh environments
  - e.g. outside in the sun, on your smartphone

It's easier to fix contrast issues early, so test while **designing**.

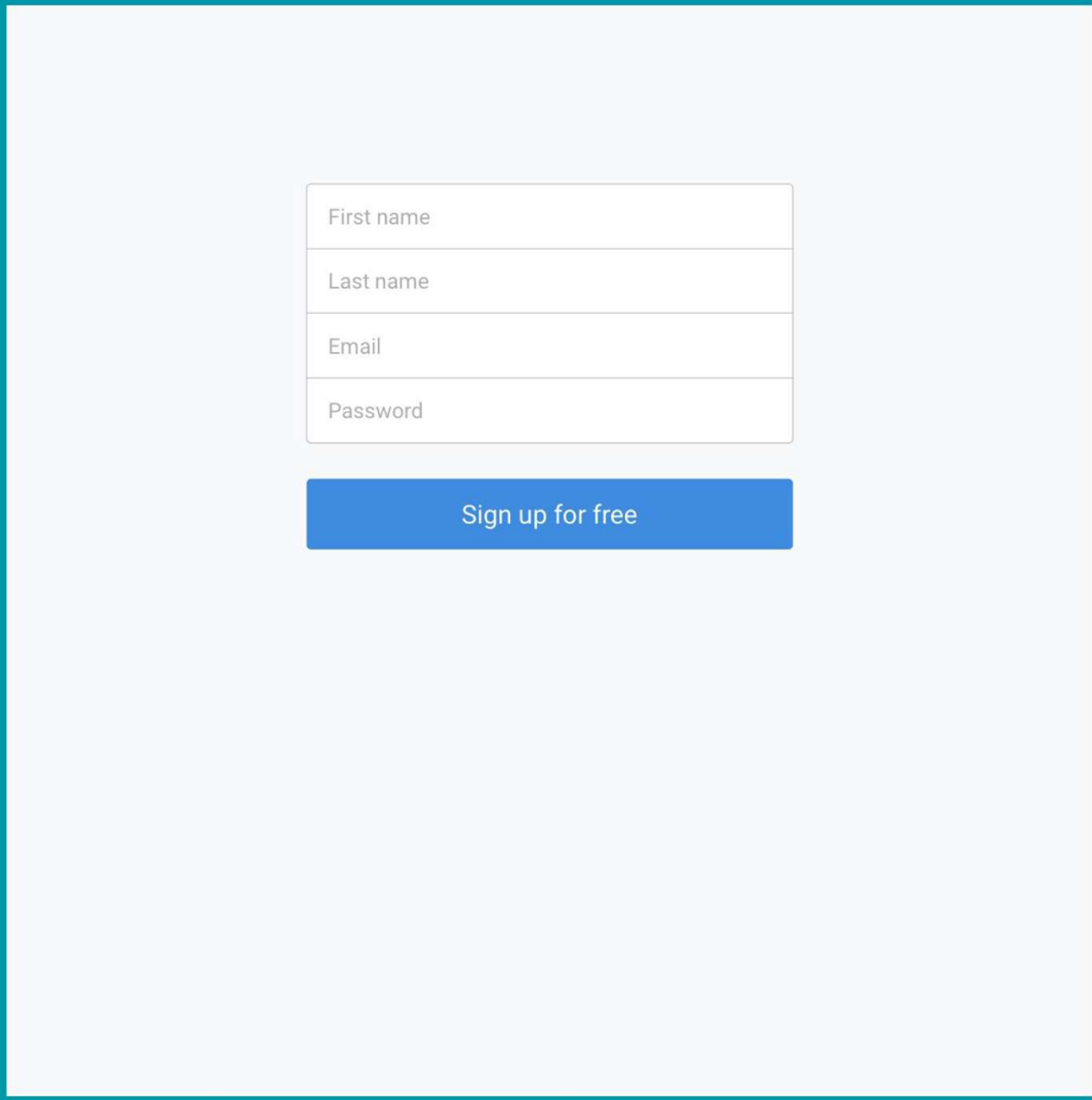

Sketch File Edit Insert Layer Text Prototype Arrange Plugins View Window Help
a11y-design
Insert Data Create Symbol Forward Backward Group Ungroup Edit Rotate Mask Scale Flatten Union Subtract Intersect Difference Assistants
LAYERS COMPONENTS
Contrast, Color, Size
Contrast Ratio
Search Layers
Accessible Colours - best
Inaccessible Colours
Accessible contrast - best
Accessible contrast - bare mi...
Inaccessible Contrast
login form
Sign up for free
button
separator 1
separator 2
separator 3
First name
Last name
Email
Password
form background
background
Inaccessible Contrast
First name
Last name
Email
Password
Sign up for free

@larenelg

Bad contrast

"First name" on background contrast ratio:

2.18:1

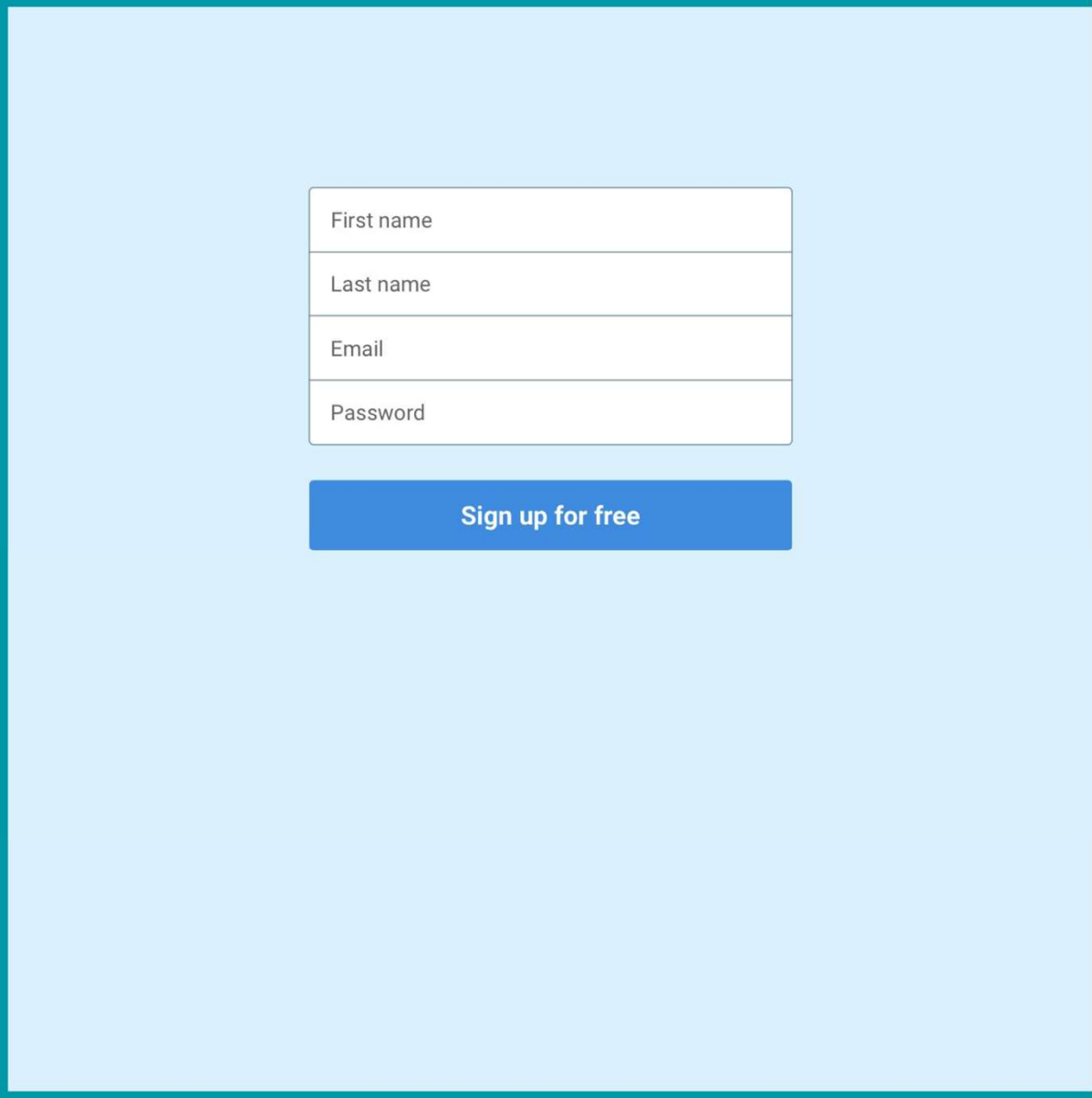


Source:

Better contrast

"First name" on background contrast ratio:

5.84:1



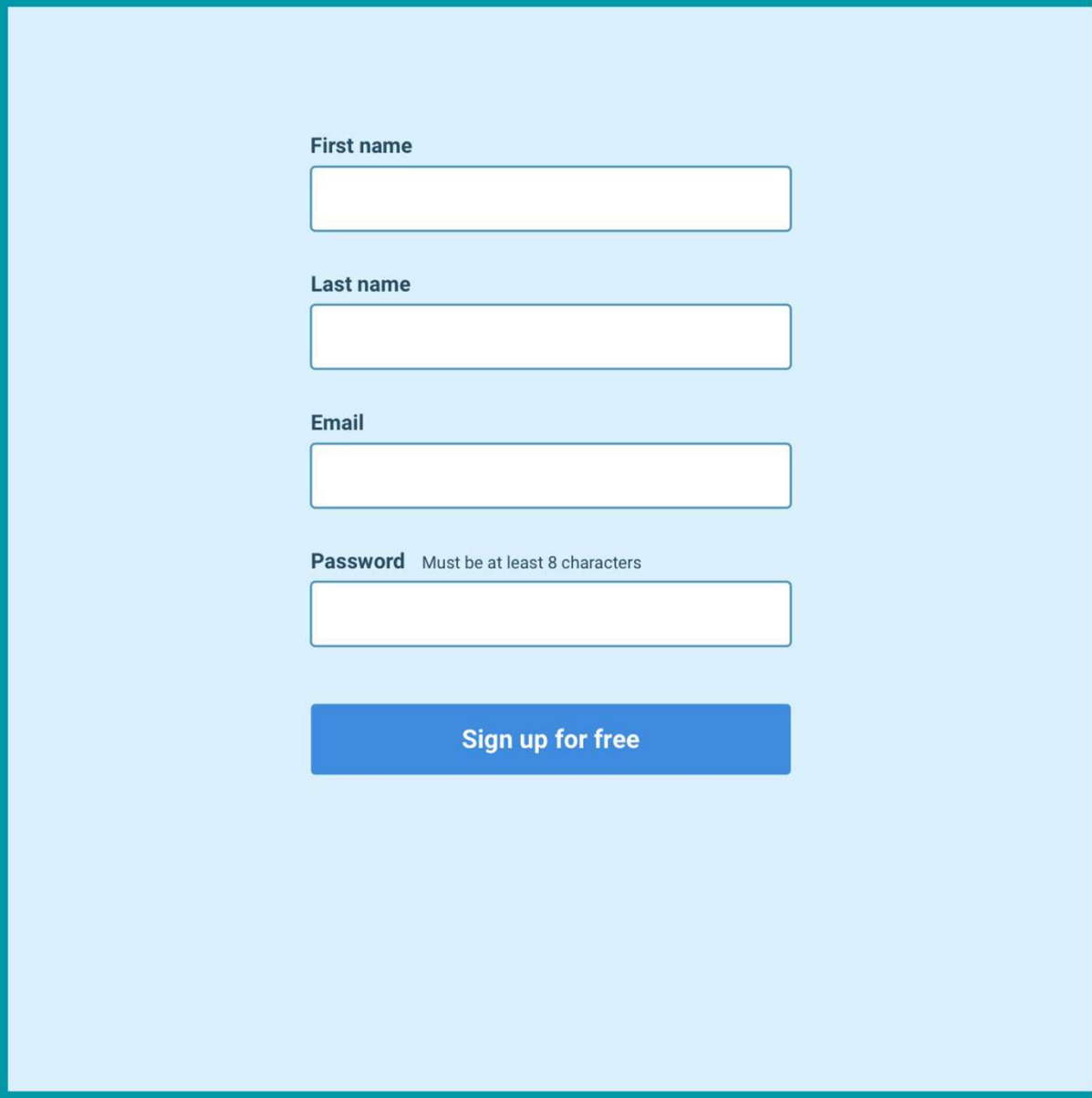
Source:

Best contrast

"First name" on background contrast ratio:

7.82:1

Source:



Colour & contrast checking tools

Colourblind simulators

- Stark plugin (Sketch app)
- colororacle.org (Mac, Win, Linux)
- https://whocanuse.com/
- find your own browser extension by searching for "color blindness simulator browser extension"

Contrast checkers

- Stark plugin (Sketch app)
- https://webaim.org/resources/contrastchecker/
- included in most Accessibility Checkers (talking about these next)

@larenelg

@LareneLG #GAAD

5 days to #a11y

1. Unplug your mouse
2. Add colour and contrast checkers to your tool belt
3. Install an accessibility checker browser extension
4.
5.

Spend 30 minutes a day this week, to build these everyday inclusive developer habits into your workflow.

@larenelg

Accessibility checkers

So, we've talked about keyboard, and colour and contrast.

Other common accessibility mistakes, such as

- images without alt-text
- unlabelled `<input>`s
- icon-only links and buttons, with no labels
- and more...

can all be found in one go with an automated **accessibility checker**.

What is an "accessibility checker"?

Automated accessibility checkers are code analysers, they are software that

- scans a web page's HTML, and
- generates a report of accessibility problems

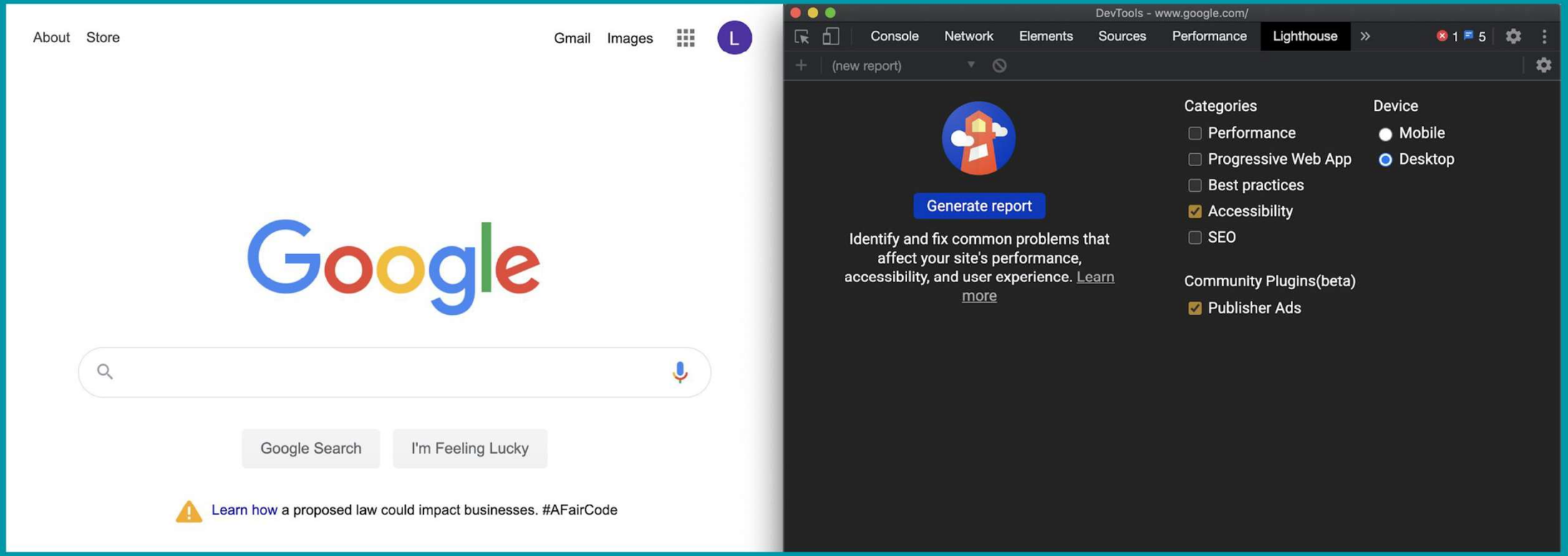
This report resembles something like what you would see out of a professional accessibility audit, with one big disclaimer....

Accessibility checkers only catch ~30% of accessibility problems.

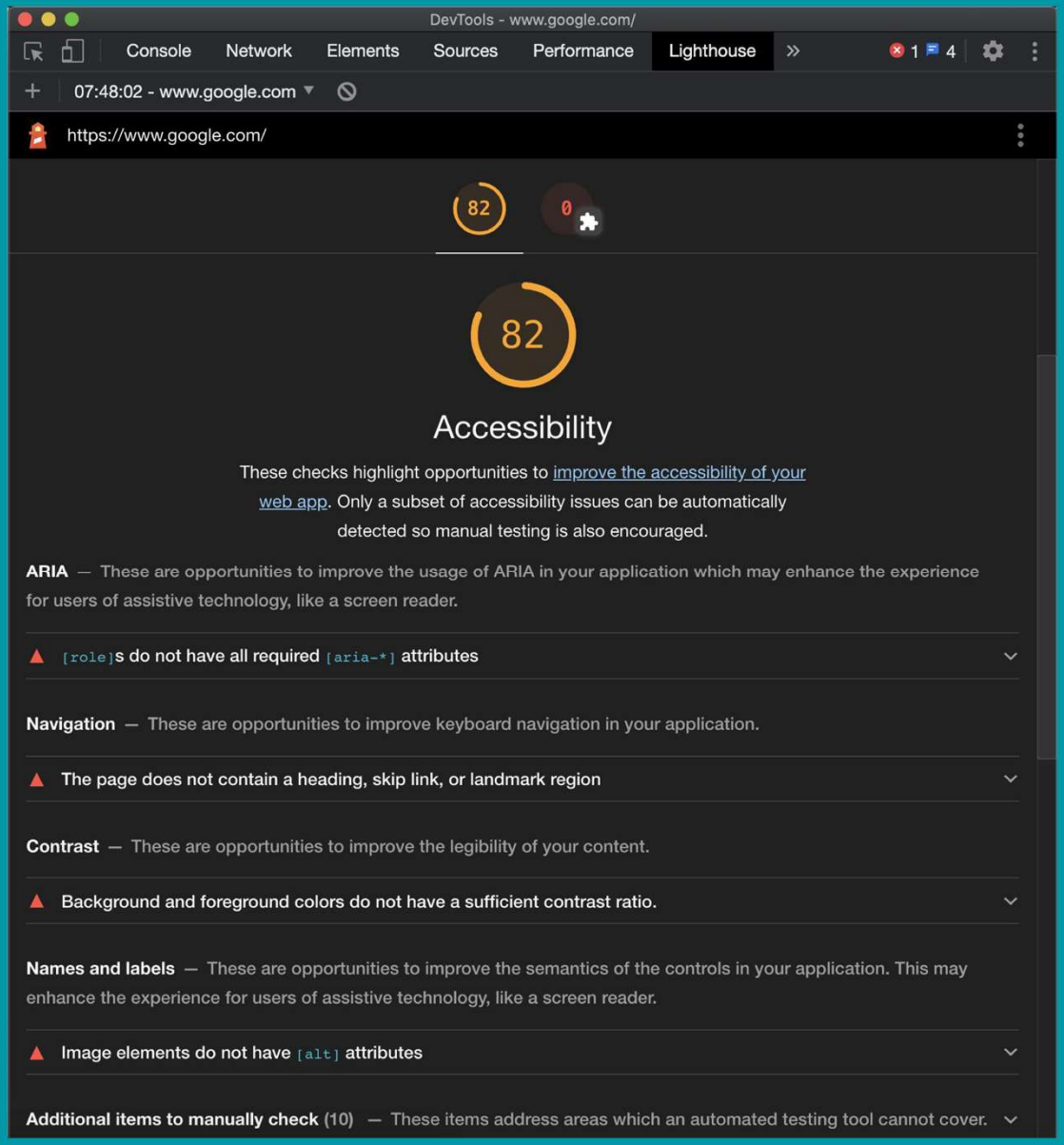
You still need to test manually, it's just a tool that helps make accessibility testing faster.

@larenelg

Source: https://alphagov.github.io/accessibility-tool-audit/


About
Store
Gmail
Images
L
Google
Google Search
I'm Feeling Lucky
Learn how a proposed law could impact businesses. #AFairCode
DevTools - www.google.com/
Console
Network
Elements
Sources
Performance
Lighthouse
1
5
(new report)
Generate report
Identify and fix common problems that affect your site's performance, accessibility, and user experience. Learn more
Categories
Performance
Progressive Web App
Best practices
Accessibility
SEO
Community Plugins(beta)
Publisher Ads
Device
Mobile
Desktop

@larenelg


DevTools - www.google.com/
Console
Network
Elements
Sources
Performance
Lighthouse
1
4
07:48:02 - www.google.com
https://www.google.com/
82
0
82
Accessibility
These checks highlight opportunities to improve the accessibility of your web app. Only a subset of accessibility issues can be automatically detected so manual testing is also encouraged.
ARIA — These are opportunities to improve the usage of ARIA in your application which may enhance the experience for users of assistive technology, like a screen reader.
[role]s do not have all required [aria-*] attributes
Navigation — These are opportunities to improve keyboard navigation in your application.
The page does not contain a heading, skip link, or landmark region
Contrast — These are opportunities to improve the legibility of your content.
Background and foreground colors do not have a sufficient contrast ratio.
Names and labels — These are opportunities to improve the semantics of the controls in your application. This may enhance the experience for users of assistive technology, like a screen reader.
Image elements do not have [alt] attributes
Additional items to manually check (10) — These items address areas which an automated testing tool cannot cover.

@larenelg

How to use accessibility checkers

Automated accessibility checkers check for compliance with WCAG (Web Content Accessibility Guidelines) standards - these are the web standards for accessibility, a checklist of sorts.

In Australia, all public sector digital products (i.e. government) must comply with WCAG 2.0 Level AA standards.

If you're a software developer, you need to be familiar with WCAG in order to build accessible products.

Start with this great WCAG 2.0 checklist:
https://webaim.org/standards/wcag/checklist

@larenelg

@LareneLG #GAAD

My #a11y browser tools

Google's Lighthouse

Deque's axe

MSFT Accessibility Insights

Spend 30 minutes a day this week, to build these everyday inclusive developer habits into your workflow.

@larenelg

I use accessibility checkers
multiple times a day
when I'm building
a new user interface

@larenelg

@LareneLG #GAAD

5 days to #a11y

1. Unplug your mouse
2. Add colour and contrast checkers to your tool belt
3. Install an accessibility checker browser extension
4. Learn to use a screen reader
5.

Spend 30 minutes a day this week, to build these everyday inclusive developer habits into your workflow.

@larenelg

Assistive Tech

JAWS® SCREEN READER

Source: Dropbox

Source: Wikipedia

NVAccess/NVDA

Apple VoiceOver

Souce: zlikovec/Shutterstock

@larenelg

Screen readers

- Assistive technology is any technology that allows people with disabilities to achieve what would otherwise be impossible
- Screen readers are a type of assistive tech
- Learning to use a screen reader is the best thing you can do to make sure your website or apps accessible
- Being able to navigate a website with a screen reader also means it is keyboard-only friendly
- I always use a screen reader while building a new user interface

@larenelg

@LareneLG #GAAD

Free screen readers I use

NVDA on Windows **VoiceOver on Mac/iOS** **TalkBack on Android**

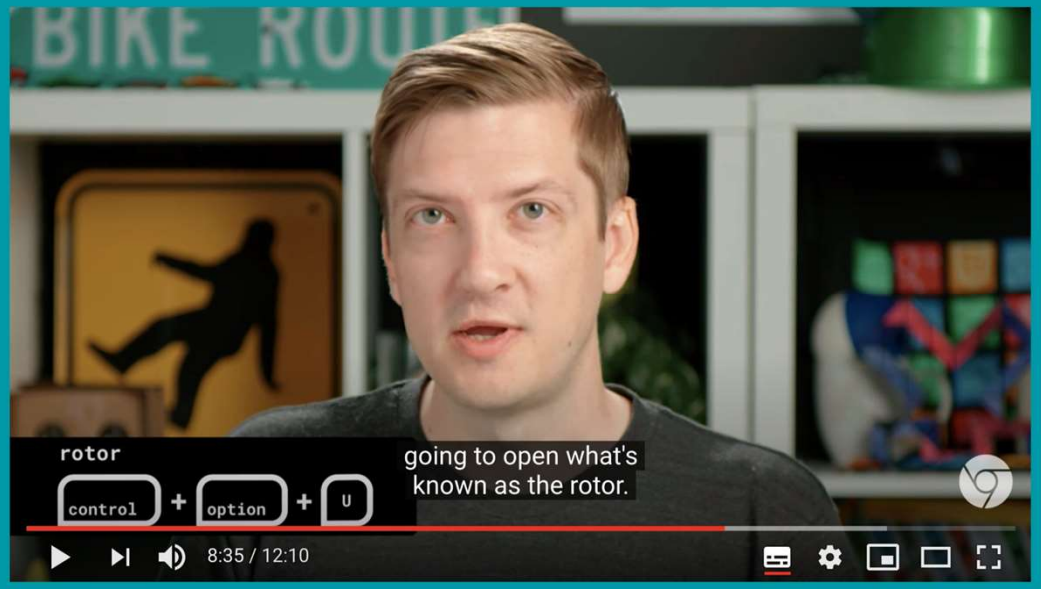
Spend 30 minutes a day this week, to build these everyday inclusive developer habits into your workflow.

@larenelg

Learn to use a screen reader in <10 minutes!

I recommend these online tutorials

- NVDA on Windows
- VoiceOver on Mac



Don't be discouraged if you don't pick it up straight away! Keep practicing. I usually have a printout of the keyboard shortcuts on my desk.

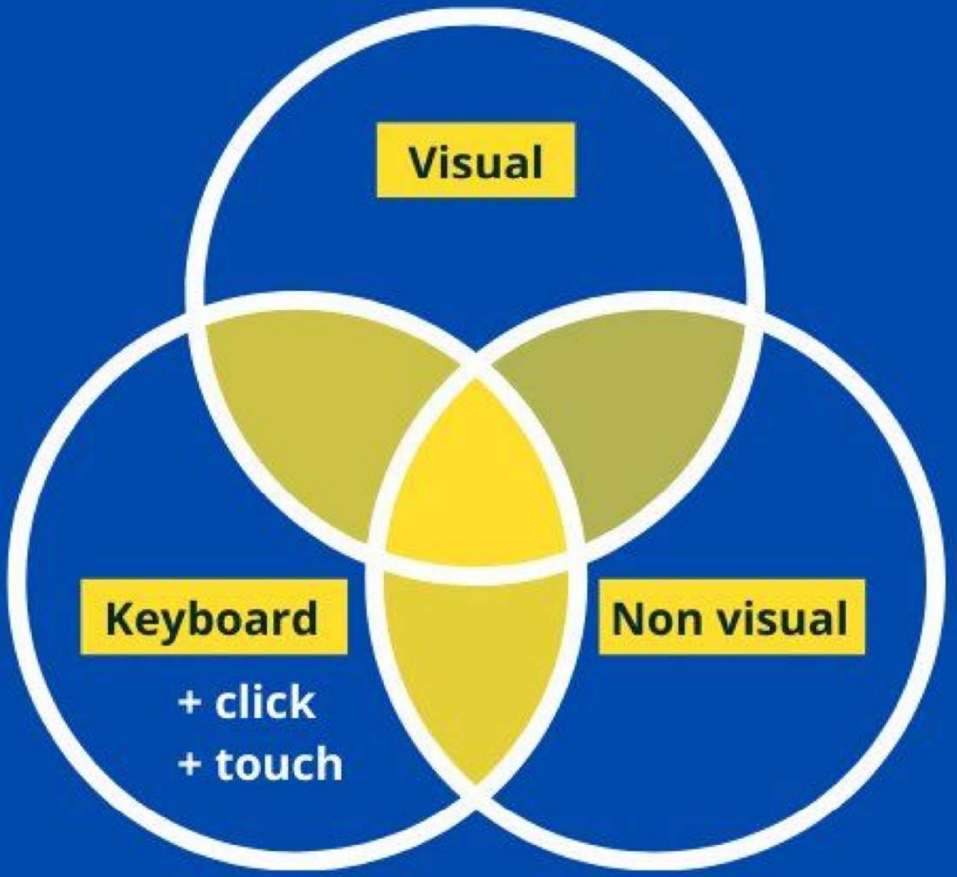
@larenelg

Only by using a screen reader,
will you experience the importance
of headings, alt-tags,
form <label>s, and many more
accessibility features!

@larenelg

@LareneLG #GAAD

How I design inclusive web components



@larenelg

Example - shopping cart button

Let's say we're designing a shopping cart "button".

- Visual users need hover, focus, and active states.
- Mouse or touch users need to be able to press it to follow the link.
- **Keyboard-only users need to be able to Tab to it, and press Enter to follow the link.**
- **Non-visual users need to know it's a hyperlink that takes them to their cart.**

If we plan for keyboard-only and non-visual, and test with colorblind, contrast, and accessibility checkers and a screen reader, we'll have a pretty good chance of building something accessible!

@larenelg

@LareneLG #GAAD

Designing an inclusive cart "button"

Visual

Keyboard

Tab

Enter

+ click
+ touch

Non-visual

ROLE link

TEXT ALT "shopping cart, 6 items"

@larenelg

@LareneLG #GAAD

5 days to #a11y

1. Unplug your mouse
2. Add colour and contrast checkers to your tool belt
3. Install an accessibility checker browser extension
4. Learn to use a screen reader
5. Continuing your accessibility learning

Spend 30 minutes a day this week, to build these everyday inclusive developer habits into your workflow.

@larenelg

So, what is "ARIA"?

- ARIA (or WAI-ARIA) stands for:
  - Web Accessibility Initiative - Accessible Rich Internet Applications
- ARIA are a collection of HTML tags created for software developers to add to websites that have lots of interactive and/or dynamic content (i.e. lots of JavaScript)
- Using semantic HTML and ARIA correctly is essential for people who use screen readers, to be able to understand what is happening on the page
- Most websites I've seen don't use semantic HTML and ARIA correctly

ARIA is confusing for many, especially if you don't use a screen reader (SR).

ARIA only adds semantics, it does NOT add behaviour.

@larenelg

<div **role="button"**>

Tells a screen reader
"this is a button"

It *does not* make it behave like a button. You need JavaScript for that.

@larenelg

The first rule of ARIA is...
Don't use ARIA, use semantic HTML

The second rule is... no ARIA is better than bad ARIA

@larenelg

We cannot build accessible websites without understanding and practicing using ARIA tags correctly.

@larenelg

I **always** refer to the ARIA Authoring Practices and Examples when I'm building a new interactive component

https://www.w3.org/TR/wai-aria-practices-1.1/examples/

@larenelg

@LareneLG #GAAD

ARIA - how to make a be a <button>

Sematic HTML

button.html

```
<button onclick="doThing()">
  Do thing
</button>
```

button.js

```
function doThing () {...}
```

All this javascript is needed to make a act the same as a native <button>? Just use a <button>!

Not using semantic HTML

div_button.html

```
<div tabindex="0" role="button" id="div_button">
  Do thing
</div>
```

div_button.js

```
var button = document.getElementById('div_button');
button.addEventListener('click', doThing);
button.addEventListener('keydown', keydownHandler);
button.addEventListener('keyup', keyupHandler);

function doThing () {...}
function keydownHandler () {...}
function keyupHandler () {...}
```

@larenelg

Things to remember

@larenelg

Don't forget about keyboard accessibility!

e.g. needs to also have tabindex="0" to let users Tab to it, on top of the extra JavaScript to add "space" and "enter" keyboard behaviour.

You don't need tabindex for native interactive elements such as <a> and <button>, it's built in. That's why it's always preferable to use semantic HTML - you get a lot of accessibility for free!

@larenelg

Most code examples are inaccessible!

- Many code snippets on the web are inaccessible!

Even in documentation of the most popular JavaScript frameworks, and trusted online courses and tutorials.

If you're copying HTML from anywhere on the web, chances are you'll need to add keyboard navigation, extra JavaScript, and ARIA to it, to make it accessible.

Finding examples of accessible components

- Google "accessible <component>" first
  - I recently googled "accessible light dark toggle" and found this amazing tutorial by Sara Soueidan: https://www.sarasoueidan.com/blog/toggle-switch-design/

There's also a great collection of accessible component guides by Heydon Pickering: https://inclusive-components.design/

In-depth accessibility course

There's a bunch to learn to be able to create accessible websites and apps.

Here's a free Udacity course from Google that I always recommend to developers new to accessibility:

https://www.udacity.com/course/web-accessibility--ud891

I've shared this with many developers in Australia, junior to senior, who highly recommend taking the time to do the course.

@larenelg

Thanks!

@larenelg