



Beyond Borders (and Outlines):

Crafting Inclusive Experiences
with Modern CSS

Manuel Sánchez

manuelsanchezdev.com

DIE  ZEIT





Facts about me

1. I studied Translation and Interpreting





Facts about me

1. I studied Translation and Interpreting
2. I did a bootcamp 5 years ago to become a Frontend Developer



Facts about me

1. I studied Translation and Interpreting
2. I did a bootcamp 5 years ago to become a Frontend Developer
3. **Work at Die Zeit as a Full Stack for almost 1 year (it's been fun!)**



Facts about me

1. I studied Translation and Interpreting
2. I did a bootcamp 5 years ago to become a Frontend Developer
3. Work at Die Zeit as a Full Stack for almost 1 year (it's been fun!)
4. I develop some games and do pixel art in my free time



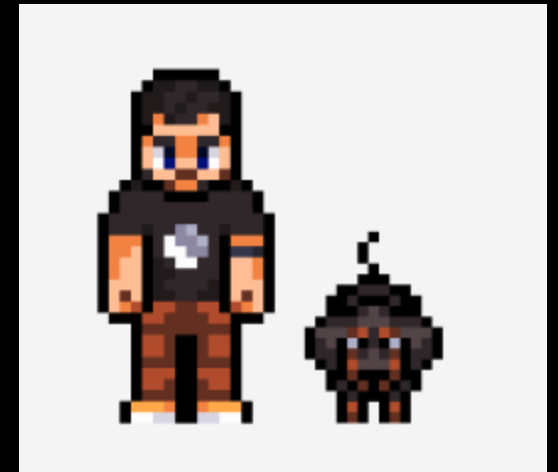
Facts about me

1. I studied Translation and Interpreting
2. I did a bootcamp 5 years ago to become a Frontend Developer
3. Work at Die Zeit as a Full Stack for almost 1 year (it's been fun!)
4. I develop some games and do pixel art in my free time
5. I love a11y!



Facts about me

1. I studied Translation and Interpreting
2. I did a bootcamp 5 years ago to become a Frontend Developer
3. Work at Die Zeit as a Full Stack for almost 1 year (it's been fun!)
4. I develop some games and do pixel art in my free time
5. I love a11y!



Crafting Inclusive Experiences with Modern CSS

**Debug A11Y Issues
with CSS**

**Get always enough
contrast with CSS**

Crafting Inclusive Experiences with Modern CSS

**Debug A11Y Issues
with CSS**

**Get always enough
contrast with CSS**

How to spot A11Y issues?

Usual Path



TOOLS

(contrast analysers,
automatic/manual tests...)




CSS

SOLUTION

How to spot A11Y issues?

Usual Path



 **TOOLS**
(contrast analysers,
automatic/manual tests...)



 **CSS**
SOLUTION

CSS First Path



  **CSS AS A**
TOOL



 **HTML / ARIA / JS**
SOLUTION

CASE 1

Debug A11Y Issues with CSS





```
<div class="btn" role="button" tabindex="0" aria-label="Open menu">  
  ≡  
</div>
```

```
<span class="btn" role="link" tabindex="0"  
  onclick="window.location='https://example.com'">  
  Read More  
</span>
```

Interactive elements

Adding a semantic role is not enough — you also need to ensure the necessary event listeners are in place, add a `tabindex`, and apply default styles.



Go to Example

HTML

```
<div class="btn" role="button" tabindex="0" aria-label="Open menu">
  ≡
</div>

<span class="btn" role="link" tabindex="0"
onclick="window.location='https://example.com'">
  Read More
</span>
```

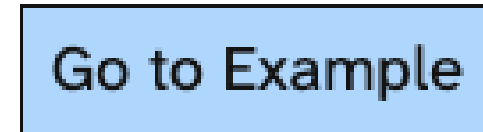
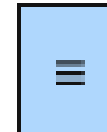
CSS

```
[role="button"],
[role="link"] {
  outline: 2px dashed var(--color-debug-warning);
  outline-offset: 4px;
  position: relative;
}
```

```
[role="button"]::after,
[role="link"]::after {
  content: " ⚠ Role needs keyboard support";
  color: var(--color-debug-warning);
  font-size: 0.675rem;
  display: block;
  border: 1px solid var(--color-dark);
  background-color: var(--color-light);
  padding: 0.25em 0.75em;
  border-radius: 1.25em;
}
```

Interactive elements

We could have just targeted **div** or **span**, but people tend to get creative.



```
<div class="btn" role="button" tabindex="0" aria-label="Open menu">
  ≡
</div>

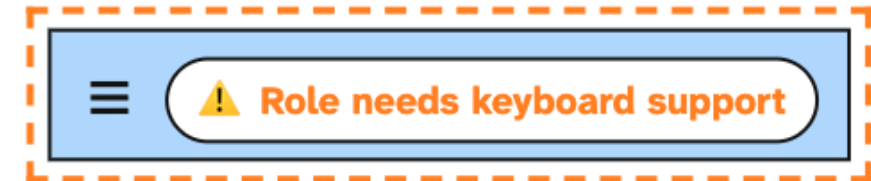
<span class="btn" role="link" tabindex="0"
onclick="window.location='https://example.com'">
  Read More
</span>
```

```
[role="button"],
[role="link"] {
  outline: 2px dashed var(--color-debug-warning);
  outline-offset: 4px;
  position: relative;
}

[role="button"]::after,
[role="link"]::after {
  content: " ⚠ Role needs keyboard support";
  color: var(--color-debug-warning);
  font-size: 0.675rem;
  display: block;
  border: 1px solid var(--color-dark);
  background-color: var(--color-light);
  padding: 0.25em 0.75em;
  border-radius: 1.25em;
}
```

Interactive elements

Now we're ready to identify those interactive elements and use native `<button>` or `<a>` tags whenever possible.



```
<table>
<tr>
  <th>Company</th>
  <th>Contact</th>
  <th>Country</th>
</tr>
<tr>
  <td>Alfreds Futterkiste</td>
  <td>Maria Anders</td>
  <td>Germany</td>
</tr>
<tr>
  <td>Centro comercial Moctezuma</td>
  <td>Francisco Chang</td>
  <td>Mexico</td>
</tr>
</table>
```

Tables without caption

Adding a <caption> tag helps satisfy Success Criterion 1.3.1: Info and Relationships.

```
table:not(:has(caption)) {  
  outline: 2px dashed var(--color-debug-error);  
  outline-offset: 4px;  
  position: relative;  
  width: 100%;  
}  
  
table:not(:has(caption))::after {  
  content: "❌ Table missing <caption>";  
  color: var(--color-dark);  
  font-size: 0.675rem;  
  font-weight: bold;  
  display: block;  
  margin-block: 0.25em;  
  background-color: var(--color-debug-light-error);  
  border: 1px solid var(--color-dark);  
  padding: 0.25em 0.75em;  
  border-radius: 1.25em;  
  width: fit-content;  
}
```

Tables without caption

Adding a <caption> tag helps satisfy Success Criterion 1.3.1: Info and Relationships.

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico
❌ Table missing <caption>		

```
<table>
<tr>
  <th>Company</th>
  <th>Contact</th>
  <th>Country</th>
</tr>
<tr>
  <td>Alfreds Futterkiste</td>
  <td>Maria Anders</td>
  <td>Germany</td>
</tr>
<tr>
  <td>Centro comercial Moctezuma</td>
  <td>Francisco Chang</td>
  <td>Mexico</td>
</tr>
</table>
```

Tables without caption

After reviewing the table, we decide to add a caption for clarity.

```
<table>
  <caption>Company Contacts</caption>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>
```

Tables without caption

Good job!

Company Contacts

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico

```
<ul>  
  <div>Wrong child</div>  
  <li>Okay</li>  
</ul>
```

```
<ol>  
  <div>Wrong child</div>  
  <li>Okay</li>  
</ol>
```

UL/OL with invalid children

```
ul:has(> :not(li)),  
ol:has(> :not(li)) {  
  outline: 2px dashed var(--color-debug-warning);  
  outline-offset: 4px;  
  position: relative;  
}
```

```
ul:has(> :not(li))::before,  
ol:has(> :not(li))::before {  
  content: " ⚠ List has children that aren't <li>";  
  color: var(--color-debug-warning);  
  font-size: 0.675rem;  
  font-weight: bold;  
  display: block;  
  margin-top: 0.25em;  
  border: 1px solid var(--color-dark);  
  background-color: var(--color-light);  
  padding: 0.25em 0.75em;  
  border-radius: 1.25em;  
  width: fit-content;  
}
```

UL/OL with invalid children

⚠ List has children that aren't

Wrong child

- Okay

⚠ List has children that aren't

Wrong child

1. Okay


```
<div>  
  <h3></h3>  
</div>
```

Empty headings

```
<div>
  <h3></h3>
</div>
```

```
h1:empty,
h2:empty,
h3:empty,
h4:empty,
h5:empty,
h6:empty {
  outline: 2px solid var(--color-debug-error);
  outline-offset: 4px;
  position: relative;
}
```

```
h1:empty::before,
h2:empty::before,
h3:empty::before,
h4:empty::before,
h5:empty::before,
h6:empty::before {
  content: " ⚠ Empty heading";
  color: var(--color-debug-warning);
  font-size: 0.675rem;
  font-weight: bold;
  display: block;
  margin-top: 0.25em;
  border: 1px solid var(--color-dark);
  background-color: var(--color-light);
  padding: 0.25em 0.75em;
  border-radius: 1.25em;
  width: fit-content;
}
```

Empty headings

```
<div>
  <h3></h3>
</div>
```

```
h1:empty,
h2:empty,
h3:empty,
h4:empty,
h5:empty,
h6:empty {
  outline: 2px solid var(--color-debug-error);
  outline-offset: 4px;
  position: relative;
}
```

```
h1:empty::before,
h2:empty::before,
h3:empty::before,
h4:empty::before,
h5:empty::before,
h6:empty::before {
  content: " ⚠ Empty heading";
  color: var(--color-debug-warning);
  font-size: 0.675rem;
  font-weight: bold;
  display: block;
  margin-top: 0.25em;
  border: 1px solid var(--color-dark);
  background-color: var(--color-light);
  padding: 0.25em 0.75em;
  border-radius: 1.25em;
  width: fit-content;
}
```

Empty headings



Empty heading

```
<div>  
    
</div>
```

```
<div>  
    
</div>
```

```
<div>  
    
</div>
```

Img without alt tag



```
<div>
```

```
  
```

```
</div>
```

```
<div>
```

```
  
```

```
</div>
```

```
<div>
```

```
  
```

```
</div>
```

Img without alt tag



```
*:has(> img:not([alt])) {  
  outline: 2px dashed var(--color-debug-error);  
  outline-offset: 4px;  
  position: relative;  
}  
  
*:has(> img:not([alt]))::after {  
  content: "❌ Image missing alt text";  
  color: var(--color-dark);  
  font-size: 0.675rem;  
  font-weight: bold;  
  display: block;  
  margin-top: 0.25em;  
  background-color: var(--color-debug-light-error);  
  border: 1px solid var(--color-dark);  
  padding: 0.25em 0.75em;  
  border-radius: 1.25em;  
  width: fit-content;  
}
```

Img without alt tag

We need to target the direct parent, since the img tag doesn't support ::after or ::before.



```
*:has(> img:not([alt])) {  
  outline: 2px dashed var(--color-debug-error);  
  outline-offset: 4px;  
  position: relative;  
}  
  
*:has(> img:not([alt]))::after {  
  content: "❌ Image missing alt text";  
  color: var(--color-dark);  
  font-size: 0.675rem;  
  font-weight: bold;  
  display: block;  
  margin-top: 0.25em;  
  background-color: var(--color-debug-light-error);  
  border: 1px solid var(--color-dark);  
  padding: 0.25em 0.75em;  
  border-radius: 1.25em;  
  width: fit-content;  
}
```

Img without alt tag

We need to address the direct parent, since img tag does not allow ::after and ::before.



```
<div>
  <label>Email address</label>
</div>

<div>
  <label>Email address</label>
  <input type="email" id="email_no_for" />
</div>

<div>
  <label for="email_no_id">Email address</label>
  <input type="email" />
</div>
```

Inputs without labels //

Labels without "for"

Email address

Email address

Email address


```
<div>
  <label>Email address</label>
</div>

<div>
  <label>Email address</label>
  <input type="email" id="email_no_for" />
</div>

<div>
  <label for="email_no_id">Email address</label>
  <input type="email" />
</div>
```

Inputs without labels //

Labels without "for"

⚠ Label missing 'for' or control

Email address

⚠ Label missing 'for' or control

Email address

Email address

```
label:not([for]):not(:has(input, select, textarea)),
label[for]+input:not([id]),
label[for]+select:not([id]),
label[for]+textarea:not([id]) {
  outline: 2px dashed var(--color-debug-error);
  outline-offset: 4px;
  position: relative;
}
```

```
label:not([for]):not(:has(input, select, textarea))::before,
label[for]+input:not([id])::before,
label[for]+select:not([id])::before,
label[for]+textarea:not([id])::before {
  content: " ⚠ Label missing 'for' or control";
  color: var(--color-debug-error);
  font-size: 0.675rem;
  font-weight: bold;
  display: block;
  margin-top: 0.25em;
  background-color: var(--color-debug-light-error);
  border: 1px solid var(--color-dark);
  padding: 0.25em 0.75em;
  border-radius: 1.25em;
  width: fit-content;
}
```

Inputs without labels //

Labels without "for"



We do not see a message in the latest case, because input does not allow pseudoelements.

```
<button class="btn">
  <svg aria-hidden="true" xmlns="http://www.w3.org/2000/svg" width="24" height="24"
  viewBox="0 0 24 24"
  fill="none" stroke="currentColor" stroke-width="2" stroke-linecap="round" stroke-
  linejoin="round"
    <path d="m22 7-8.991 5.727a2 2 0 0 1-2.009 0L2 7" />
    <rect x="2" y="4" width="20" height="16" rx="2" />
  </svg>
</button>
```

(Icon) Button with no aria-label



```
button:has(svg):not(:has(span)):not([aria-label]) {  
  outline: 2px dashed var(--color-debug-warning);  
  outline-offset: 4px;  
  position: relative;  
}  
  
button:has(svg):not(:has(span)):not([aria-label])::after {  
  content: " ⚠ Icon-only button needs a label";  
  color: var(--color-debug-error);  
  font-size: 0.675rem;  
  font-weight: bold;  
  display: block;  
  margin-top: 0.25em;  
  background-color: var(--color-debug-light-error);  
  border: 1px solid var(--color-dark);  
  padding: 0.25em 0.75em;  
  border-radius: 1.25em;  
  width: fit-content;  
}
```

(Icon) Button with no aria-label

You can even enforce rules — for example, if a button contains an svg, the text must be wrapped in a span.

(You can define different warnings or errors as needed.)



CASE 2

Get always enough
contrast with CSS



LEA VEROU



On compliance vs readability: Generating text colors with CSS

📅 17 May 2024

🕒 18 min read

@ in 🐦 📧

ON THIS PAGE

Relative Colors

The CSS `contrast-color()` function

Must read: <https://lea.verou.me/blog/2024/contrast-color/>

- Lea Verou, the CSS Queen



Must read: <https://lea.verou.me/blog/2024/contrast-color/>

- Lea Verou, the CSS Queen
- Worked at W3C as a Dev Advocate



LEA VEROU

On compliance vs readability: Generating text colors with CSS

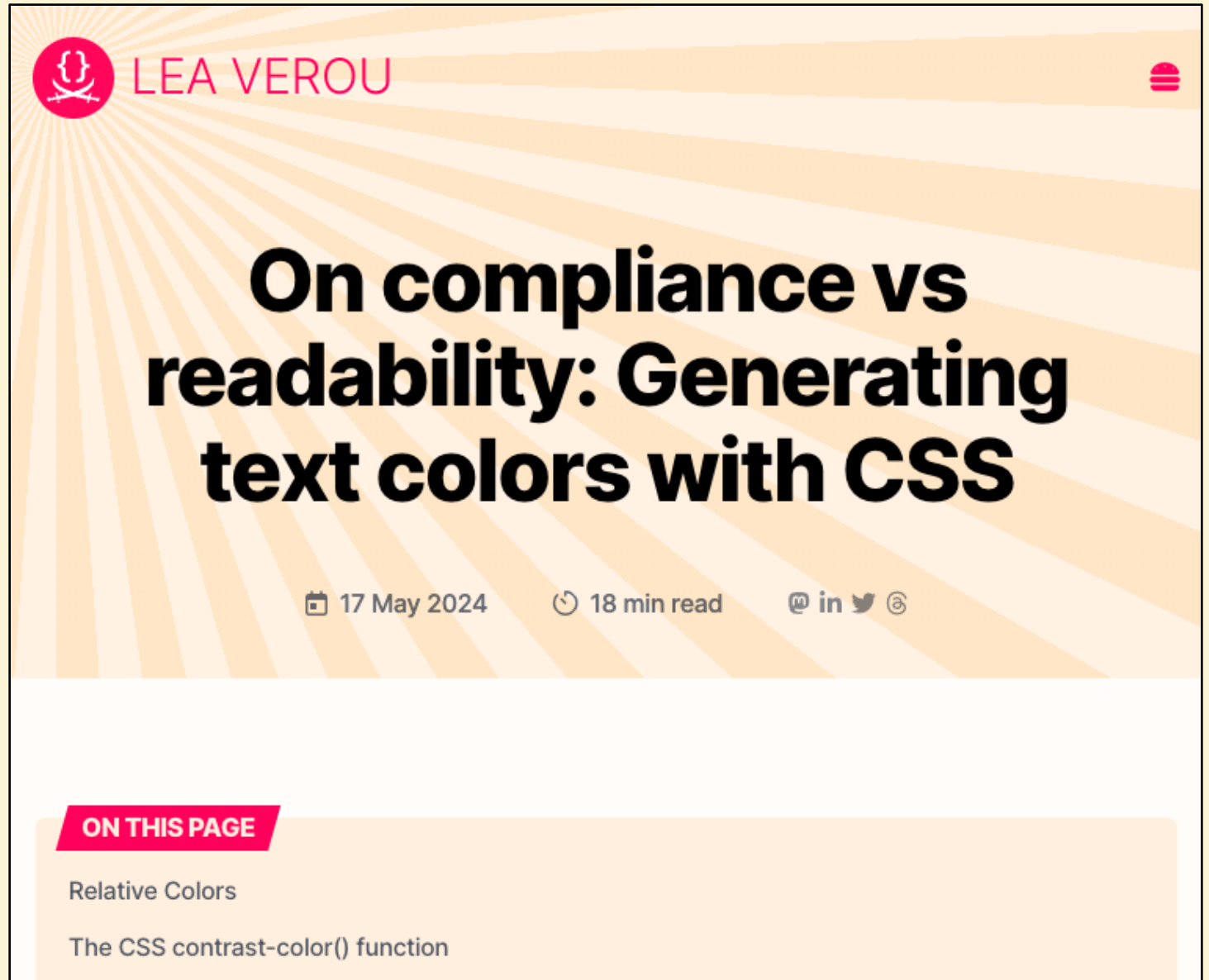
📅 17 May 2024 ⌚ 18 min read @ in 🐦 📧

ON THIS PAGE

- Relative Colors
- The CSS `contrast-color()` function

Must read: <https://lea.verou.me/blog/2024/contrast-color/>

- Lea Verou, the CSS Queen
- Worked at W3C as a Dev Advocate
- Many open source projects (like PrismJS or Color.js)



Must read: <https://lea.verou.me/blog/2024/contrast-color/>

- Lea Verou, the CSS Queen
- Worked at W3C as a Dev Advocate
- Many open source projects (like PrismJS or Color.js)
- Involved in many proposals like :is(), :has() or if() and *Relative Color Syntax(RCS)*

```
--color-lighter: hsl(from var(--color) h s calc(1 * 1.2));  
--color-lighterer: oklch(from var(--color) calc(1 + 0.2) c h);  
--color-alpha-50: oklab(from var(--color) l a b / 50%);
```

The header features a pink circular profile picture of Lea Verou and her name 'LEA VEROU' in pink. The background has a pattern of radiating orange and yellow lines. The title 'On compliance vs readability: Generating text colors with CSS' is in large, bold black font. Below the title, it says '17 May 2024', '18 min read', and social media icons for @, in, and a speech bubble. A pink tab labeled 'ON THIS PAGE' is visible, with a list of topics: 'Relative Colors' and 'The CSS contrast-color() function'.

LEA VEROU

On compliance vs readability: Generating text colors with CSS

📅 17 May 2024 ⌚ 18 min read @ in 🗨

ON THIS PAGE

- Relative Colors
- The CSS contrast-color() function

Must read: <https://lea.verou.me/blog/2024/contrast-color/>

contrast-color()

contrast-color()

```
background: var(--color);  
color: contrast-color(var(--color));
```

```
background: var(--color);  
color: contrast-color(var(--color));
```

contrast-color()

```
:root {  
  --color: #FFF;  
}
```

```
background: var(--color);  
color: contrast-color(var(--color));
```

```
:root {  
  --color: #000;  
}
```

```
background: var(--color);  
color: contrast-color(var(--color));
```

contrast-color()

```
:root {  
  --color: #FFF;  
}
```

```
background: var(--color);  
color: contrast-color(var(--color));
```

This is the text color

```
:root {  
  --color: #000;  
}
```

```
background: var(--color);  
color: contrast-color(var(--color));
```

This is the text color

When is this coming?

- The contrast-color() feature reached MVP in Level 5 (<https://github.com/w3c/csswg-drafts/issues/9166>)

When is this coming?

- The `contrast-color()` feature reached MVP in Level 5 (<https://github.com/w3c/csswg-drafts/issues/9166>)
- However, we can see it in the Editor's Draft in Level 6 (<https://drafts.csswg.org/css-color-6/#funcdef-color-layers>)

CSS Color Module Level 6

Editor's Draft, 4 March 2025



▼ More details about this document

This version:
<https://drafts.csswg.org/css-color-6/>

Latest published version:
<https://www.w3.org/TR/css-color-6/>

Feedback:
[CSSWG Issues Repository](#)
[Inline In Spec](#)

Editors:
[Chris Lilley](#) (W3C)
[Una Kravets](#) (Google)
[Lea Verou](#) (Invited Expert)
[Adam Argyle](#) (Google)

Suggest an Edit for this Spec:
[GitHub Editor](#)

Delta Spec:
yes

Test Suite:
<https://wpt.fyi/results/css/css-color/>

 Chrome 137 102/293	 Firefox 139 102/293	 Safari 18 102/293	 Edge 137 102/293
--	---	---	--

► Not Ready For Implementation

Copyright © 2025 World Wide Web Consortium. W3C® liability, trademark and permissive document license rules apply.

When is this coming?

- The `contrast-color()` feature reached MVP in Level 5 (<https://github.com/w3c/csswg-drafts/issues/9166>)
- However, we can see it in the Editor's Draft in Level 6 (<https://drafts.csswg.org/css-color-6/#funcdef-color-layers>)
- Level 5 is for feature that build on top of previous levels (3 adds new features and 4 refines them)
- Level 6 is for more advanced or experimental features.
- As I understand it, in **Level 6**, it evolved into a **general-purpose contrast function**, giving developers more control over what fallback colors to consider — which is amazing for design systems and accessibility.

CSS Color Module Level 6

Editor's Draft, 4 March 2025



▼ More details about this document

This version:
<https://drafts.csswg.org/css-color-6/>

Latest published version:
<https://www.w3.org/TR/css-color-6/>

Feedback:
[CSSWG Issues Repository](#)
[Inline In Spec](#)

Editors:
[Chris Lilley](#) (W3C)
[Una Kravets](#) (Google)
[Lea Verou](#) (Invited Expert)
[Adam Argyle](#) (Google)

Suggest an Edit for this Spec:
[GitHub Editor](#)

Delta Spec:
yes

Test Suite:
<https://wpt.fyi/results/css/css-color/>

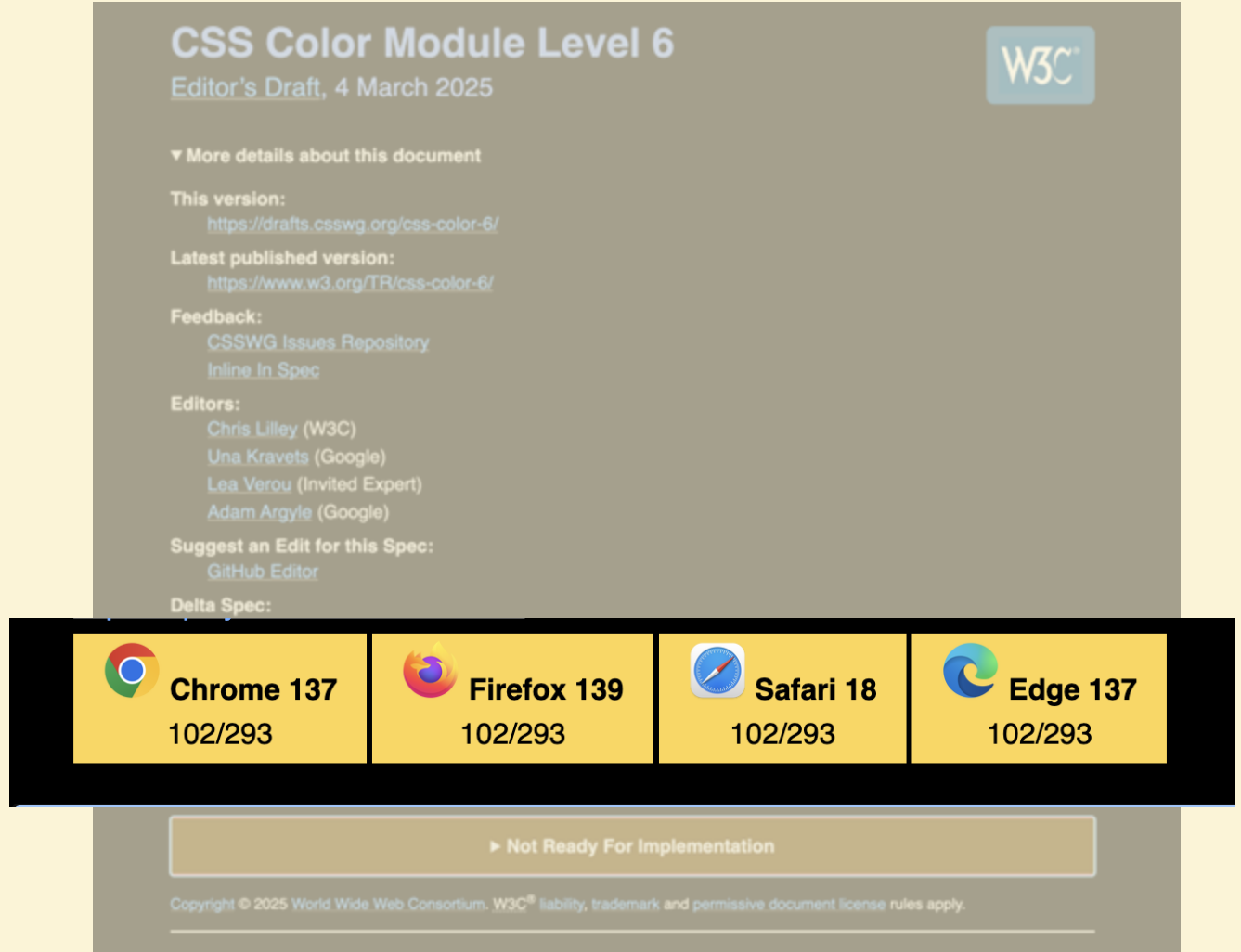
 Chrome 137 102/293	 Firefox 139 102/293	 Safari 18 102/293	 Edge 137 102/293
--	---	---	--

► Not Ready For Implementation

Copyright © 2025 World Wide Web Consortium. W3C® liability, trademark and permissive document license rules apply.

When is this coming?

- The `contrast-color()` feature reached MVP in Level 5 (<https://github.com/w3c/csswg-drafts/issues/9166>)
- However, we can see it in the Editor's Draft in Level 6 (<https://drafts.csswg.org/css-color-6/#funcdef-color-layers>)
- Level 5 is for feature that build on top of previous levels (3 adds new features and 4 refines them)
- Level 6 is for more advanced or experimental features.
- As I understand it, in **Level 6**, it evolved into a **general-purpose contrast function**, giving developers more control over what fallback colors to consider — which is amazing for design systems and accessibility.



CSS Color Module Level 6
Editor's Draft, 4 March 2025

W3C

▼ More details about this document

This version:
<https://drafts.csswg.org/css-color-6/>





Latest published version:
<https://www.w3.org/TR/css-color-6/>

Feedback:
[CSSWG Issues Repository](#)
[Inline In Spec](#)

Editors:
[Chris Lilley](#) (W3C)
[Una Kravets](#) (Google)
[Lea Verou](#) (Invited Expert)
[Adam Argyle](#) (Google)

Suggest an Edit for this Spec:
[GitHub Editor](#)

Delta Spec:

 Chrome 137 102/293	 Firefox 139 102/293	 Safari 18 102/293	 Edge 137 102/293
--	---	---	--

► Not Ready For Implementation

Copyright © 2025 World Wide Web Consortium. W3C® liability, trademark and permissive document license rules apply.

In the meantime...



```
:root {  
  /* Default color, a purple */  
  --color: oklch(65% 0.15 270);  
}
```

Step 1 – Choose your color

We will be using oklch color space.

Lightness: 65% - This determines how light or dark the color is.

Chroma: 0.15 - This indicates the saturation of the color, with lower values being less saturated.

Hue: 270 - This represents the color's position on the color wheel, with 270 degrees corresponding to a shade of purple.

```
:root {  
  /* // A threshold value for lightness */  
  --l-threshold: 62.3%;  
  
  /* Default color, a purple */  
  --color: oklch(65% 0.15 270);  
}
```

Step 2 – Compare the bg lightness to a threshold

We then compare the lightness of the background color to a threshold value (in this case, 62.3%). Why this number? After many tests with thousands of colors, Lea Verou found out that:

- **White text** always passes WCAG AA when the lightness of the bg is less than 62.3%
- **Black text** always passes WCAG AA when the lightness of the bg is greater than 62.3%

```

:root {
  /* // A threshold value for lightness */
  --l-threshold: 62.3%;

  /* Default color, a purple */
  --color: oklch(65% 0.15 270);
}

.contrast-text {
  /* Fallback */
  color: white;
  text-shadow: 0 0 0.05em black, 0 0 0.05em black;

  @supports (color: oklch(from red l c h)) {
    --l: clamp(0, (l / var(--l-threshold) - 1) * -100000, 1);
    color: oklch(from var(--color) var(--l) 0 h);
    text-shadow: none;
  }
}

```

Step 3 – We get black/white with clamp()

We can use the clamp() function to determine the text color based on the background color's lightness.

- if the background is darker than the threshold, we will get 1 as output (meaning white text)
- Otherwise, we will get 0 and therefore black text.

If the browser **does not** support this trick with oklch (**92.13% as of June 2025**), it will use a white text color with a black text shadow as a fallback.

```
:root {  
  /* // A threshold value for lightness */  
  --l-threshold: 62.3%;  
  
  /* Default color, a purple */  
  --color: oklch(65% 0.15 270);  
}  
  
.contrast-text {  
  /* Fallback */  
  color: white;  
  text-shadow: 0 0 0.05em black, 0 0 0.05em black;  
  
  @supports (color: oklch(from red l c h)) {  
    --l: clamp(0, (l / var(--l-threshold) - 1) * -100000, 1);  
    color: oklch(from var(--color) var(--l) 0 h);  
    text-shadow: none;  
  }  
  
  @supports (color: contrast-color(red)) {  
    color: contrast-color(var(--color));  
  }  
}
```

Step 3 (Part 2) – We add support for the future

And also, when one day the contrast-color() function is widely supported, we can use it directly to get the compliant text color based on the background color.

This text color adapts based on background color

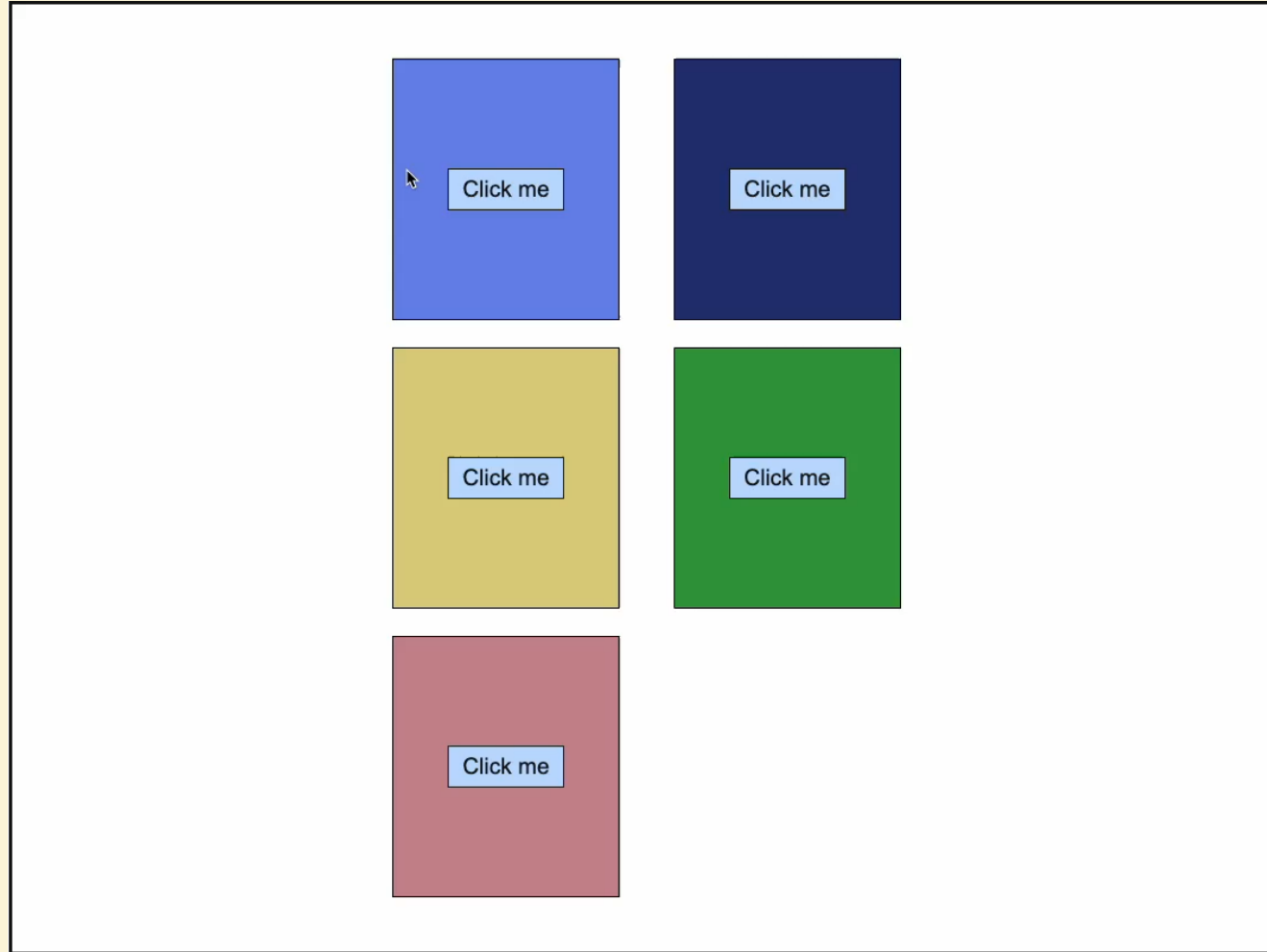
This text color adapts based on background color

This text color adapts based on background color

This text color adapts based on background color

This text color adapts based on background color

And what about outlines!



Conclusion

We can use **CSS as a tool** in addition to as a solution to fix some A11Y issues.

Be able to have **contrast-colors()** someday out of the box will help lots of people, so let's stay tuned!

References

- Lea Verou's blog article titled "On compliance vs readability: Generating text colors with CSS" : <https://lea.verou.me/blog/2024/contrast-color/>
- The contrast-color() feature reached MVP in Level 5: (<https://github.com/w3c/csswg-drafts/issues/9166>)
- Editor's Draft in Level 6 (<https://drafts.csswg.org/css-color-6/#funcdef-color-layers>)
- Repo: <https://github.com/manuelsanchez2/cssday-2025-a11y>
- Live Page: <https://cssday-2025.manuelsanchezdev.com/>

Special thanks

- Manuel Matuzović, who helped me get some presentation ideas
- Kateryna Porshnieva, who presents in such a cool way and I used some of her slide layout.

Thank you!

Find a summary of this talk and some examples under
<https://cssday-2025.manuelsanchezdev.com/>

Manuel Sánchez

manuelsanchezdev.com

DIE  ZEIT