Declarative Shadow DOM

and the future of Drupal Theming

November 17, 2025

John Albin Wilkins
Drupal & React Frontend Developer

Who is JohnAlbin?

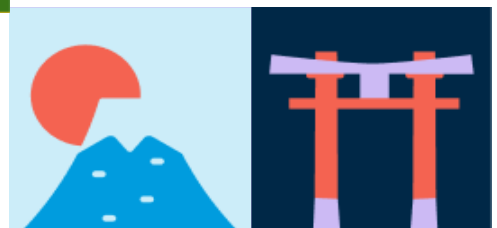
- Web developer since 1993
- Drupal dev since 2004
- Drupal core dev since 2006
- MAINTAINERS.txt since 2010
- 60+ presentations and keynotes in 4 continents

DrupalCon
Nara 2025
17-19 November

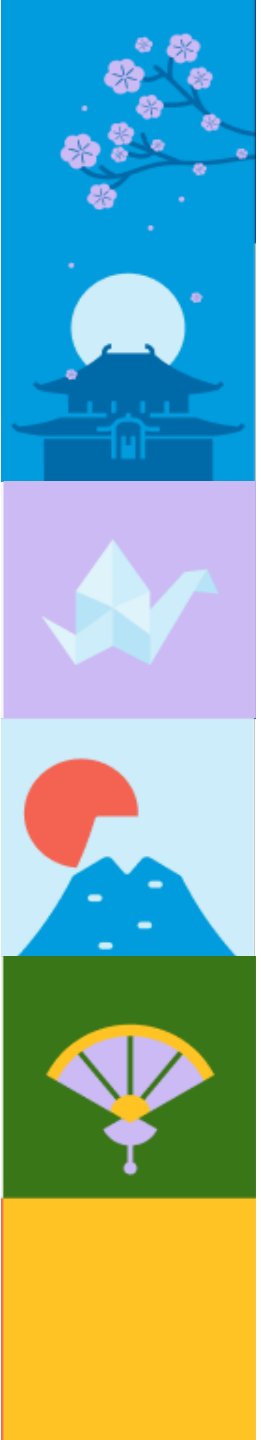
Why now?

This is my **2nd**
web components presentation.

The first was in **2014**.

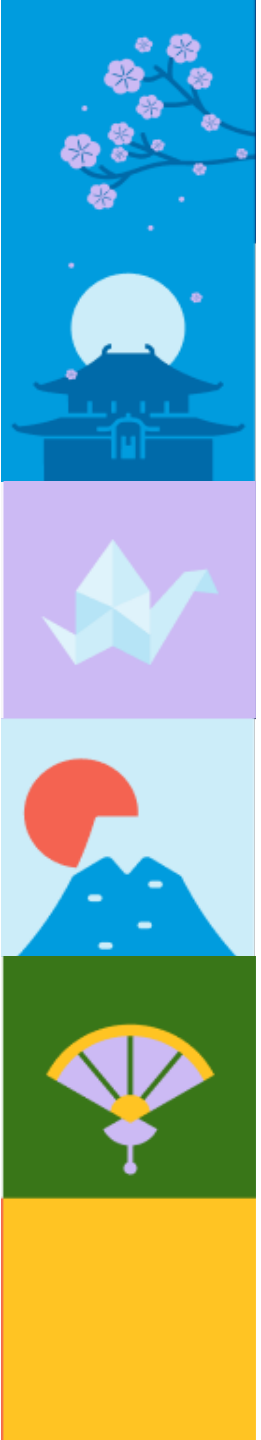


DrupalCon
Nara 2025
17-19 November



THE ERA OF REACT.JS IS COMING TO AN END

DrupalCon
Nara 2025
17-19 November



NO JAVASCRIPT FRAMEWORK WILL EVER BE AS FAST AS THE WEB PLATFORM

DrupalCon
Nara 2025
17-19 November

Baseline

Limited availability

Newly available August 5, 2024

Widely available February 5, 2027

Everything old is *new*

Old School:
PHP server-side renders

New School:
React.js' new
Server Components
and Client Components

Newer School:
PHP server-side renders

DrupalCon
Nara 2025
17-19 November

Split components

aka. progressive enhancement

1. **Server part**
   (with no client-side JavaScript)

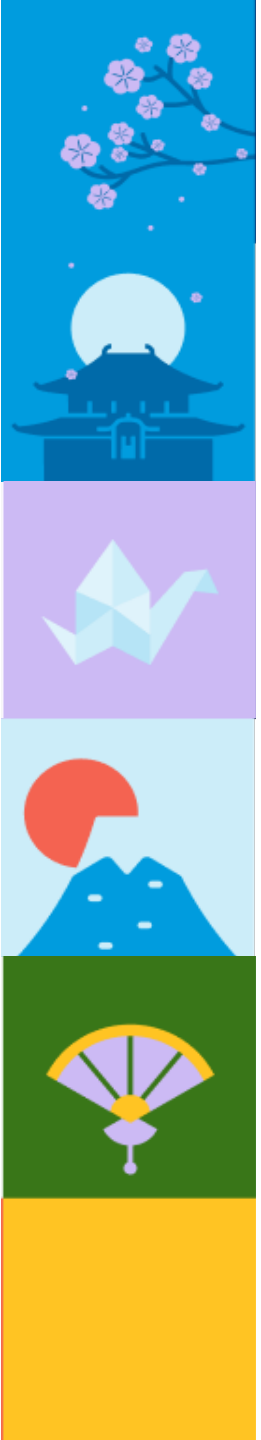
2. **Client part**
   (with client-side JavaScript)

Split components (2023)

React.js	Web Components
Server Components	*no solution*
Client Components	vanilla JavaScript

Split components (2024)

React.js	Web Components
Server Components	vanilla HTML w/ Declarative Shadow DOM
Client Components	vanilla JavaScript

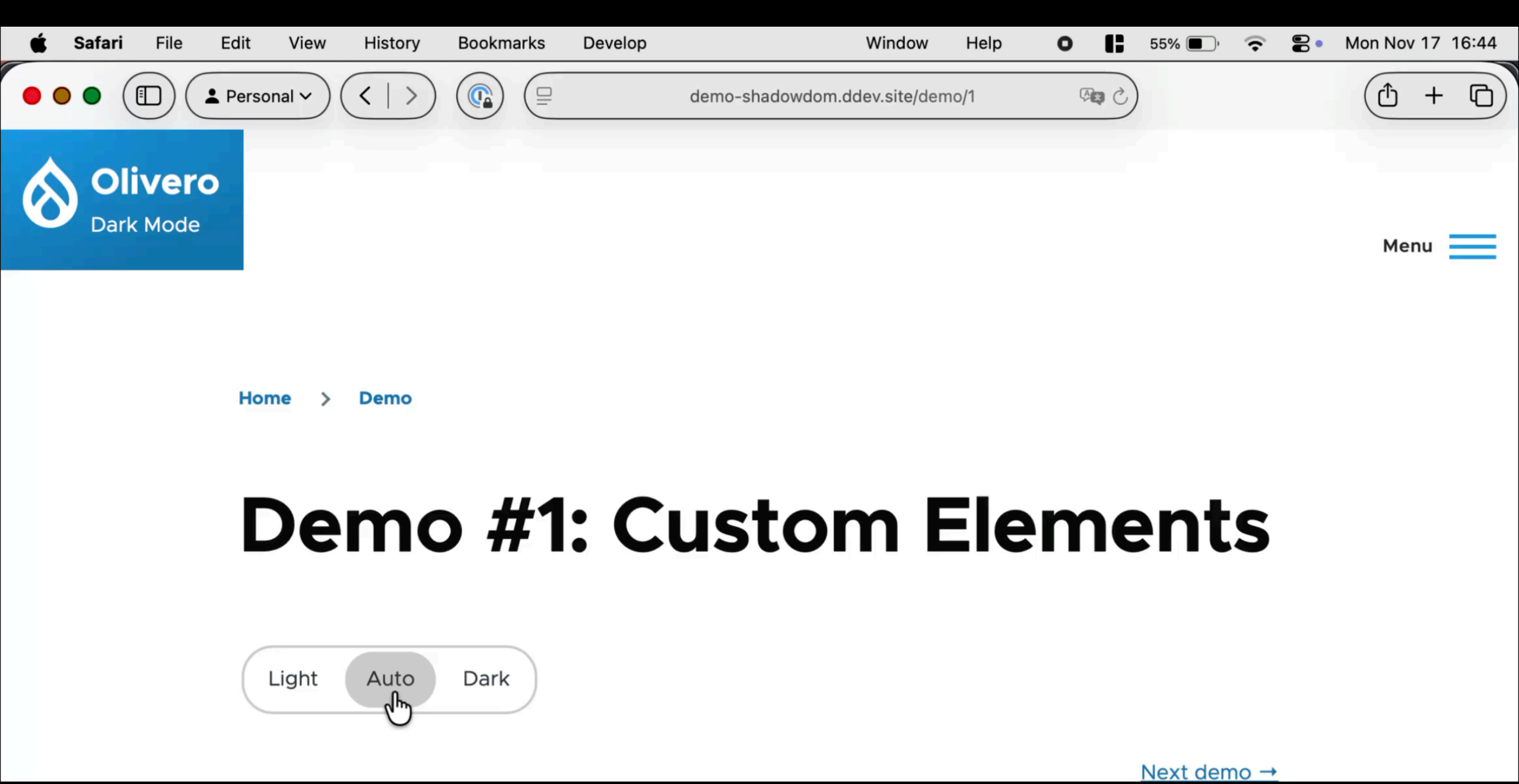


Web components

95% working LIVE DEMO!

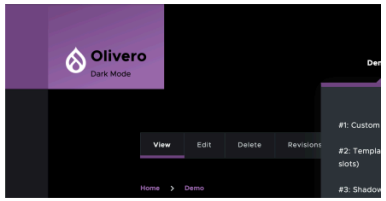
1. Custom elements
2. Templates
3. Shadow DOM

DrupalCon
Nara 2025
17-19 November


Safari File Edit View History Bookmarks Develop Window Help 55% Mon Nov 17 16:44
Personal
demo-shadowdom.ddev.site/demo/1
Olivero
Dark Mode
Menu
Home > Demo
Demo #1: Custom Elements
Light
Auto
Dark
Next demo →

Olivero Dark Mode

View | Version control



A sub-theme of Olivero that responds to your operating system's dark mode. This theme supports Olivero's ability to change its primary color too.

This theme uses the Newly Available Baseline 2024 CSS color function, `light-dark()`. This color function is specifically designed for dark mode color schemes and *will* be the way web developers implement this feature. However, this color function won't be considered Widely Available until November 2027.

Project information

Seeking co-maintainer(s)
Maintainers are looking for help reviewing issues.

Ecosystem: Olivero

23 sites report using this theme

Created by johnalbin on 4 October 2025, updated 14 October 2025

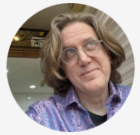
Stable releases for this project are covered by the security advisory policy. Look for the shield icon below.

Releases

1.0.1 released 20 October 2025

1 star

Maintainers



johnalbin

Issues for Olivero Dark Mode

To avoid duplicates, please search before submitting a new issue.

Search

Advanced search

All issues
1 open, 4 total

Bug report

toggle-color-scheme.twig

```twig
<toggle-color-scheme>
  <fieldset>
    <legend class="visually-hidden">
      Color theme
    </legend>
    <label>
      <input type="radio" name="toggle-color-scheme" value="light" class="visually-hidden">
      {{ 'Light'|t }}
    </label>
    <label>
      <input type="radio" name="toggle-color-scheme" value="light dark" checked class="visually-hidden">
      {{ 'Auto'|t }}
    </label>
    <label>
      <input type="radio" name="toggle-color-scheme" value="dark" class="visually-hidden">
      {{ 'Dark'|t }}
    </label>
  </fieldset>
</toggle-color-scheme>
```

toggle-color-scheme.twig | toggle-color-scheme.js | toggle-color-scheme.css

```
9  html[data-color-scheme="dark"] {
10   color-scheme: dark;
11 }
12
13 toggle-color-scheme {
14   display: block;
15
16   fieldset {
17     padding: 3px;
18     border: 2px solid #ccc;
19     border-radius: 24px;
20     inline-size: fit-content;
21   }
22
23   label {
24     display: inline-flex;
25     padding: 0.5em 1em;
26     cursor: pointer;
27     user-select: none;
28     text-align: center;
29     border-radius: 21px;
30
31     &:has(input:checked) {
32       color: #333;
33       background-color: #ccc;
34     }
```

toggle-color-scheme.twig | toggle-color-scheme.js | toggle-color-scheme.css

```
Show usages
1   class ToggleColorScheme extends HTMLElement {
2     storageKey : string  = 'toggle-color-scheme';
3
      Show usages
4     constructor() {
5       super();
6     }
7
      no usages
8     connectedCallback() : void {...}
24
      Show usages
25    setColorScheme(colorScheme) : void {...}
38  }
39
40  customElements.define( name: 'toggle-color-scheme', ToggleColorScheme);
```

toggle-color-scheme.twig | toggle-color-scheme.js | toggle-color-scheme.css

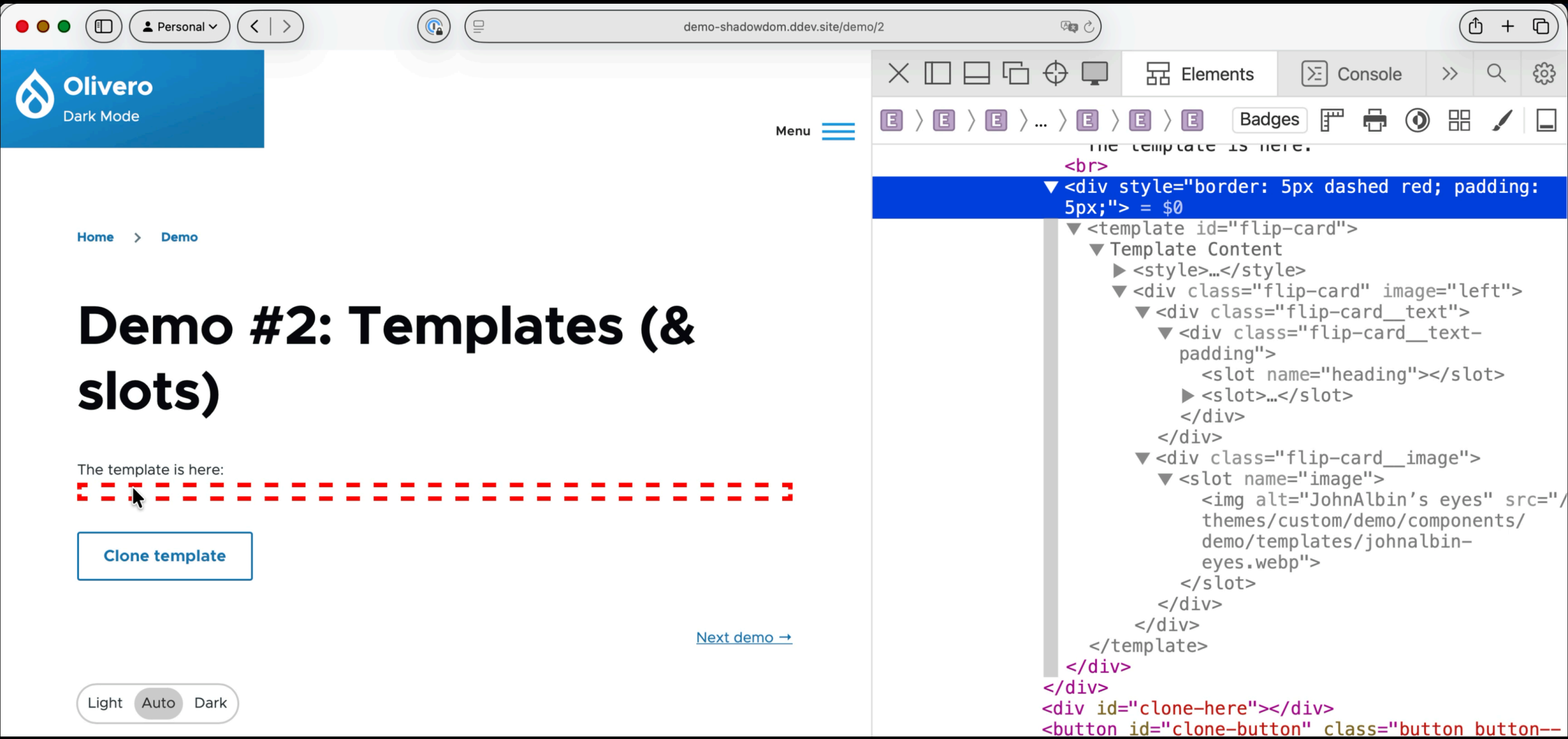
```
Show usages
class ToggleColorScheme extends HTMLElement {
  storageKey : string  = 'toggle-color-scheme';

  Show usages
  constructor() {...}

  Show usages
  connectedCallback() : void  {
    // Initialize the color scheme.
    const colorScheme : string  = localStorage.getItem(this.storageKey) ?? 'light dark';
    this.setColorScheme(colorScheme);

    // Update the form controls.
    if (colorScheme === 'light' || colorScheme === 'dark') {...}

    // Have the fieldset catch bubbling change events from the radio buttons.
    const element : ToggleColorScheme  = this;
    this.querySelector( selectors: 'fieldset').addEventListener( type: 'change',  listener: (event : Event )
      : void  => {...});
  }

  Show usages
  setColorScheme(colorScheme) : void  {...}
```

customElements.define()

means no more

Drupal.behaviors *or*

Drupal.once()


Personal
demo-shadowdom.ddev.site/demo/2
Olivero
Dark Mode
Menu
Home
Demo
Demo #2: Templates (& slots)
The template is here:
Clone template
Next demo →
Light
Auto
Dark
Elements
Console
Badges

 = $0
<template id="flip-card">
Template Content
<style>…</style>

<slot name="heading"></slot>
<slot>…</slot>

<slot name="image">
<img alt="JohnAlbin's eyes" src="/themes/custom/demo/components/demo/templates/johnalbin-eyes.webp">
</slot>

</template>

<button id="clone-button" class="button button--

templates.twig templates.js

```
3         <div style="...">
5           <div style="...">
7         <template id="flip-card">
41          <div class="flip-card" image="left">
42            <div class="flip-card__text">
43              <div class="flip-card__text-padding">
44                <slot name="heading"></slot>
45                <slot>
46                  <p>This component requires the default slot to be used.</p>
47                </slot>
48              </div>
49            </div>
50
51            <div class="flip-card__image">
52              <slot name="image">
53                <img alt="JohnAlbin's eyes" src="{{ default_image }}" />
54              </slot>
55            </div>
```

Home > Demo

Demo #3: Shadow DOM

Flip Card

Some text for the default slot.

A link that is part of the Shadow DOM.

FLIP

FLIP CARD

A flip card that doesn't specify any slots.

Look at how this link is styled

A link that is part of the Shadow DOM.

FLOP

```
page--full.html.twig ■ node--page.html.twig ■
node--full.html.twig ■ node.html.twig -->
<!-- 💡 BEGIN CUSTOM TEMPLATE OUTPUT from
'themes/custom/demo/templates/node--3.html.twig'
-->
<!-- ⚪ Component start: demo:shadow-dom -->
▶ <template id="flip-card">…</template> = $0
▼ <flip-card> grid
  ▶ Shadow Content (Open)
    <p>Some text for the default slot.</p> Slotted
    <img slot="image" alt="JohnAlbin's eyes" src=
    "/themes/custom/demo/components/demo/shadow-
    dom/johnalbin-eyes.webp"> Slotted
    <h2 slot="heading">Flip Card</h2> Slotted
  </flip-card>
▶ <flip-card image="left">…</flip-card> grid
  <!-- ⚪ Component end: demo:shadow-dom -->
  <!-- 🥓 Component start: demo:next-demo -->
▶ <div class="next-demo">…</div>
  <!-- 🥓 Component end: demo:next-demo -->
  <!-- END CUSTOM TEMPLATE OUTPUT from
  'themes/custom/demo/templates/node--3.html.twig'
  -->
</div>
</div>
```

shadow-dom.js shadow-dom.twig

```
1   class FlipCard extends HTMLElement {
2     static observedAttributes : string[]  = ['image'];
3
4     button;
5
6     /** From MDN's "Using custom elements": ...*/
      Show usages
14    constructor() {
15      super();
16
17      // Attach Shadow DOM.
18      this.attachShadow( init: { mode: 'open' });
19    }
20
      Show usages
21    connectedCallback() : void  {...}
42
      no usages
43    attributeChangedCallback(name, oldValue, newValue) : void  {...}
48
      Show usages
49    setButtonText(imageAttribute) : void  {...}
55  }
56
57  customElements.define( name: 'flip-card', FlipCard);
```

shadow-dom.js shadow-dom.twig

```
1     class FlipCard extends HTMLElement {
21      connectedCallback() : void  {
22        // Grab the template by its ID.
23        const template = document.getElementById( elementId: 'flip-card').content;
24
25        // Clone the template into the Shadow.
26        this.shadowRoot.appendChild(template.cloneNode( deep: true));
27
28        // Find the button and store a reference to it.
29        this.button = this.shadowRoot.querySelector( selectors: 'button');
30
31        // Set the initial button text.
32        this.setButtonText(this.getAttribute( qualifiedName: 'image'));
33
34        // Add an event listener.
35        this.button.addEventListener('click', () : void  => {
36          this.setAttribute(
37            qualifiedName: 'image',
38            value: this.getAttribute( qualifiedName: 'image') === 'left' ? 'right' :
               'left',
39          );
40        });
41      }
42
```

1

shadow-dom.js | shadow-dom.twig

```
<template id="flip-card">
  <link rel="stylesheet" href="{{ demo_css_path }}/shadow-root-base.css" />
  <style>
    :host {...}

    :host([image="left"]) .text {...}

    .text-padding {...}

    .image {...}

    button {...}
  </style>

  <div class="text" part="text"...>

  <div class="image" part="image"...>
</template>
```

shadow-dom.js shadow-dom.twig

```
85
86          <flip-card>
87            <p>Some text for the default slot.</p>
88            <img slot="image" alt="JohnAlbin's eyes" src="{{ image }}" />
89            <h2 slot="heading">Flip Card</h2>
90          </flip-card>
91
92          <flip-card image="left">
93            <p>A flip card that doesn't specify any slots.</p>
94            <p><a href="/">Look at how this link is styled</a></p>
95          </flip-card>
96
97          <template id="flip-card">
98            [styles...]
99            <div class="text" part="text">
100             <div class="text-padding">
101               <slot name="heading" part="heading"></slot>
102               <slot...>
105               <a href="/">A link that is part of the Shadow DOM.</a>
106             </div>
107           </div>
108
109           <div class="image" part="image">
110             <slot name="image"...>
113           </div>
```

Personal

demo-shadowdom.ddev.site/demo/3

Demo #3: Shadow DOM

Some text for the default slot.

Flip Card

Elements

Core Features

- [ ] Disable Images
- [ ] Disable CSS
- [x] Disable JavaScript

Media

- [ ] Allow media capture on insecure sites
- [ ] Disable ICE candidate restrictions
- [ ] Use mock capture devices

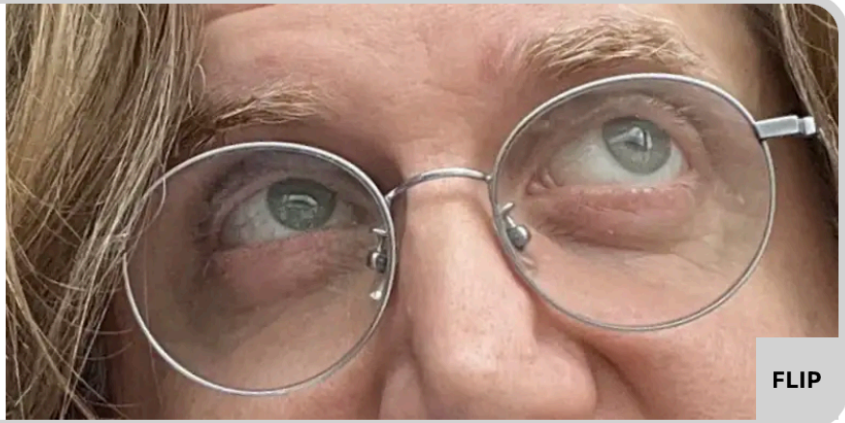
```
<template id="flip-card">…</template>
<flip-card>
  <p>Some text for the default slot.</p>
  <img slot="image" alt="JohnAlbin's eyes" src=
  "/themes/custom/demo/components/demo/shadow-
  dom/johnalbin-eyes.webp">
  <h2 slot="heading">Flip Card</h2>
</flip-card>
<flip-card image="left">…</flip-card>
<!-- Component end: demo:shadow-dom -->
```

Demo #4: Declarative Shadow DOM

Flip Card

Some text for the default slot.

A link that is part of the Shadow DOM.



FLIP

FLIP CARD

A flip card that doesn't specify any slots.

Look at how this link is styled

A link that is part of the Shadow DOM.

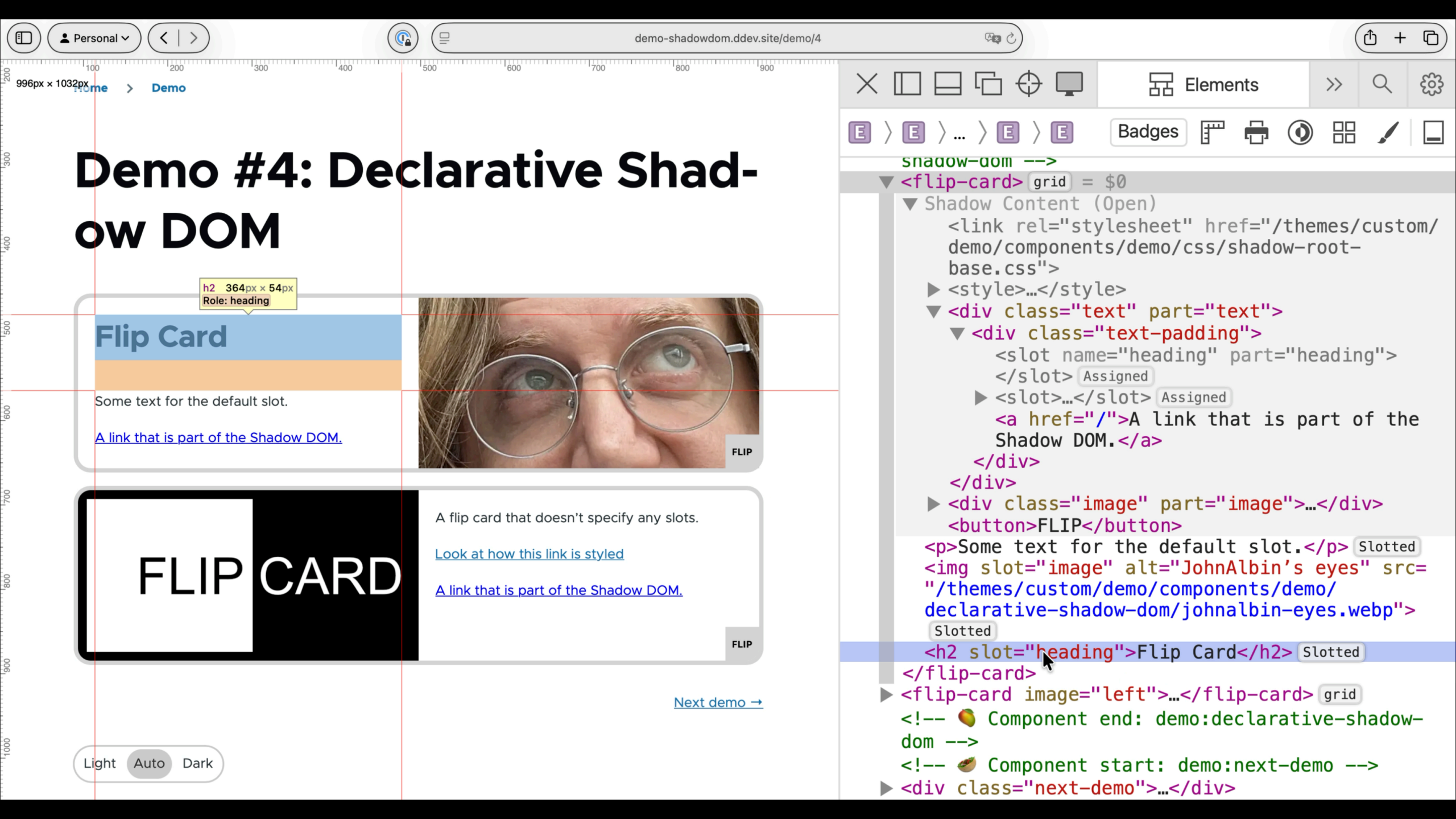
FLIP

Next demo →


Project
components
demo
css
declarative-shadow-dom
declarative-shadow-dom.comp
declarative-shadow-dom.css
declarative-shadow-dom.js
declarative-shadow-dom.twig
flip-card.webp 2025/10/9, 22:37,
johnalbin-eyes.webp 2025/10/2,
declarative-shadow-dom.js
declarative-shadow-dom.twig
1 console.log('No JavaScript was run on this page.');
2

declarative-shadow-dom.js declarative-shadow-dom.twig

```
4   <flip-card>
5     <template shadowrootmode="open" shadowrootclonable="true">
6       <link rel="stylesheet" href="{{ demo_css_path }}/shadow-root-base.css" />
7       <style...>
71
72      <div class="text" part="text">
73        <div class="text-padding">
74          <slot name="heading" part="heading"></slot>
75          <slot...>
78        </div>
79      </div>
80      <div class="image" part="image"...>
85      <button>FLIP</button>
86    </template>
87
88    <p>Some text for the default slot.</p>
89    <img slot="image" alt="JohnAlbin's eyes" src="{{ image }}" />
90    <h2 slot="heading">Flip Card</h2>
91  </flip-card>
```


Personal
demo-shadowdom.ddev.site/demo/4
996px × 1032px
Home > Demo
Demo #4: Declarative Shadow DOM
h2 364px × 54px
Role: heading
Flip Card
Some text for the default slot.
A link that is part of the Shadow DOM.
FLIP
FLIP CARD
A flip card that doesn't specify any slots.
Look at how this link is styled
A link that is part of the Shadow DOM.
FLIP
Next demo →
Light Auto Dark
Elements
Badges
shadow-dom -->
<flip-card> grid = $0
Shadow Content (Open)
<link rel="stylesheet" href="/themes/custom/demo/components/demo/css/shadow-root-base.css">
<style>…</style>

<slot name="heading" part="heading">
</slot> Assigned
<slot>…</slot> Assigned
A link that is part of the Shadow DOM.

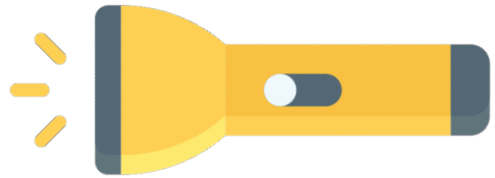
…
<button>FLIP</button>
Some text for the default slot. Slotted
<img slot="image" alt="JohnAlbin's eyes" src="/themes/custom/demo/components/demo/declarative-shadow-dom/johnalbin-eyes.webp">
Slotted
<h2 slot="heading">Flip Card</h2> Slotted
</flip-card>
<flip-card image="left">…</flip-card> grid
<!-- Component end: demo:declarative-shadow-dom -->
<!-- Component start: demo:next-demo -->
…

Shadow DOM + CSS

DrupalCon Nara 2025 17-19 November

Shadow DOM

Light DOM

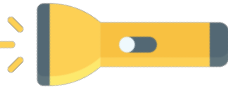


Shadow DOM + CSS

DrupalCon Nara 2025 17-19 November

<custom-element>

</body>

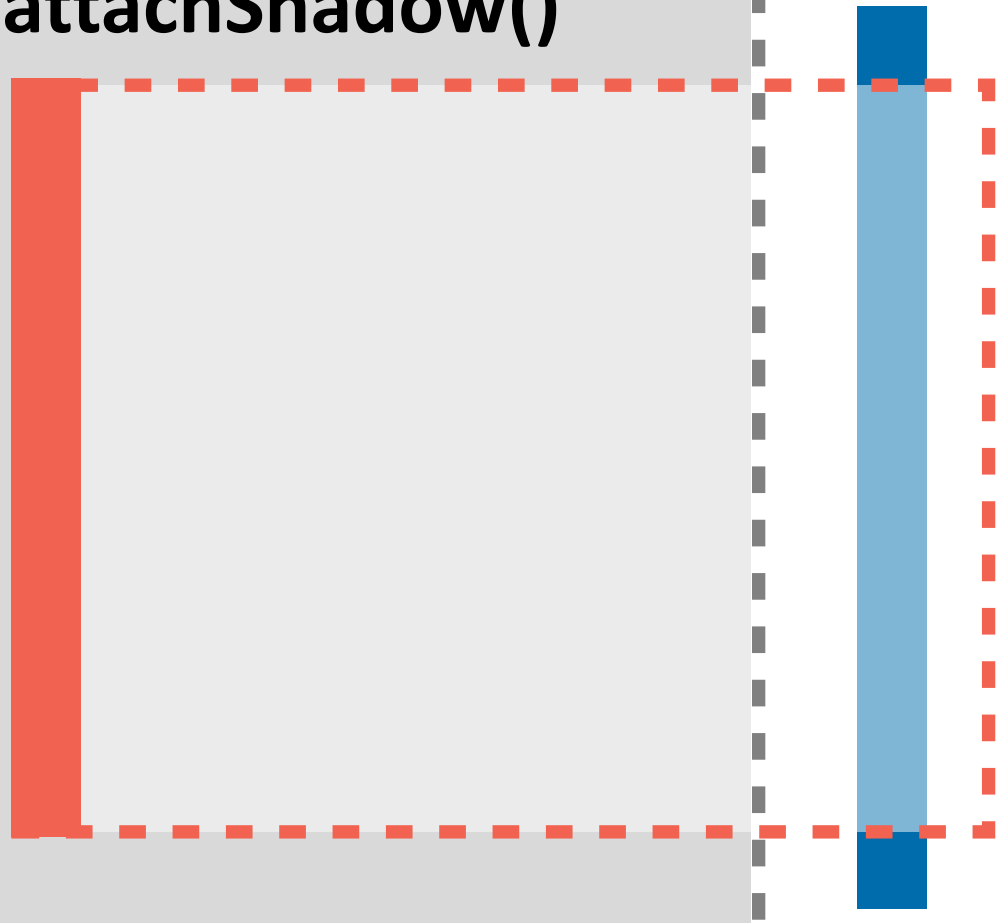


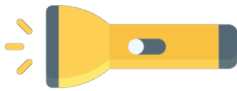
Shadow DOM

Light DOM

Shadow DOM + CSS

Call .attachShadow()

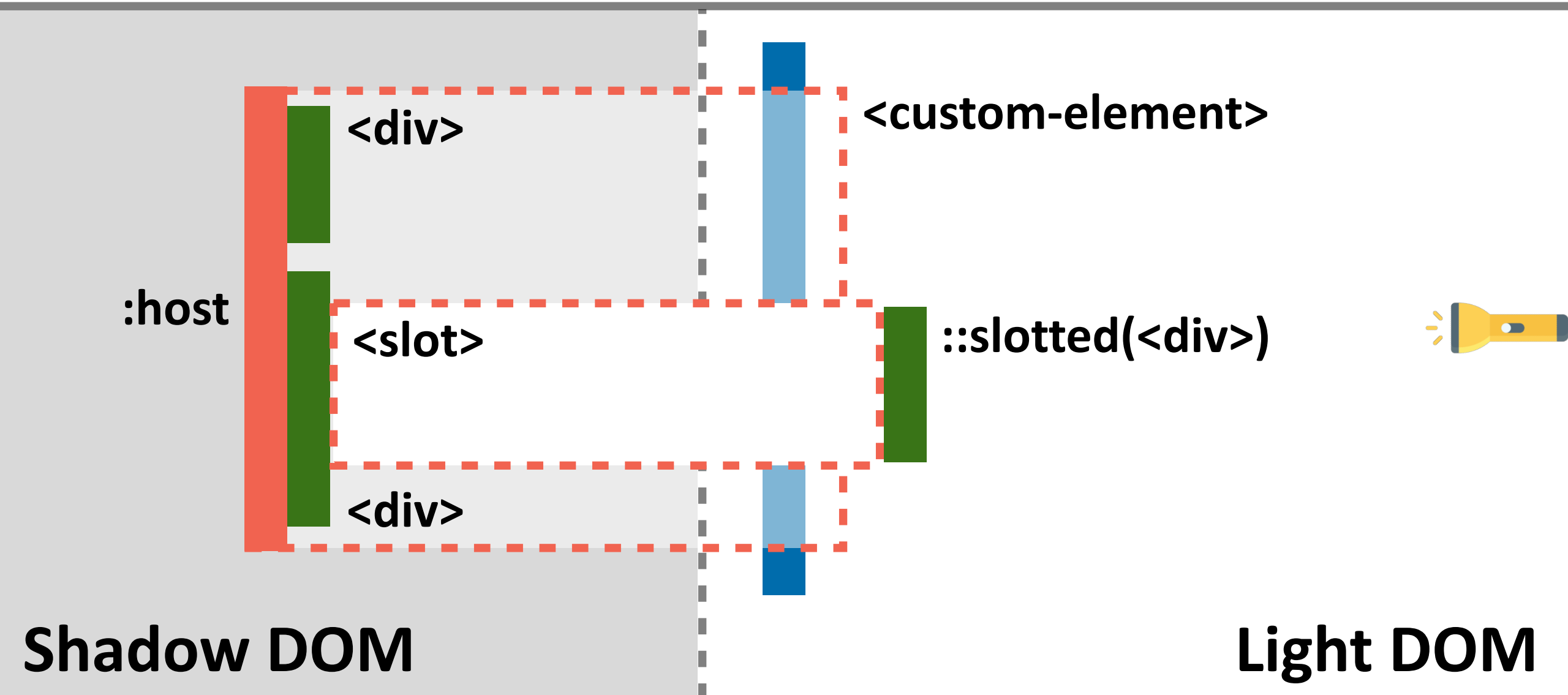




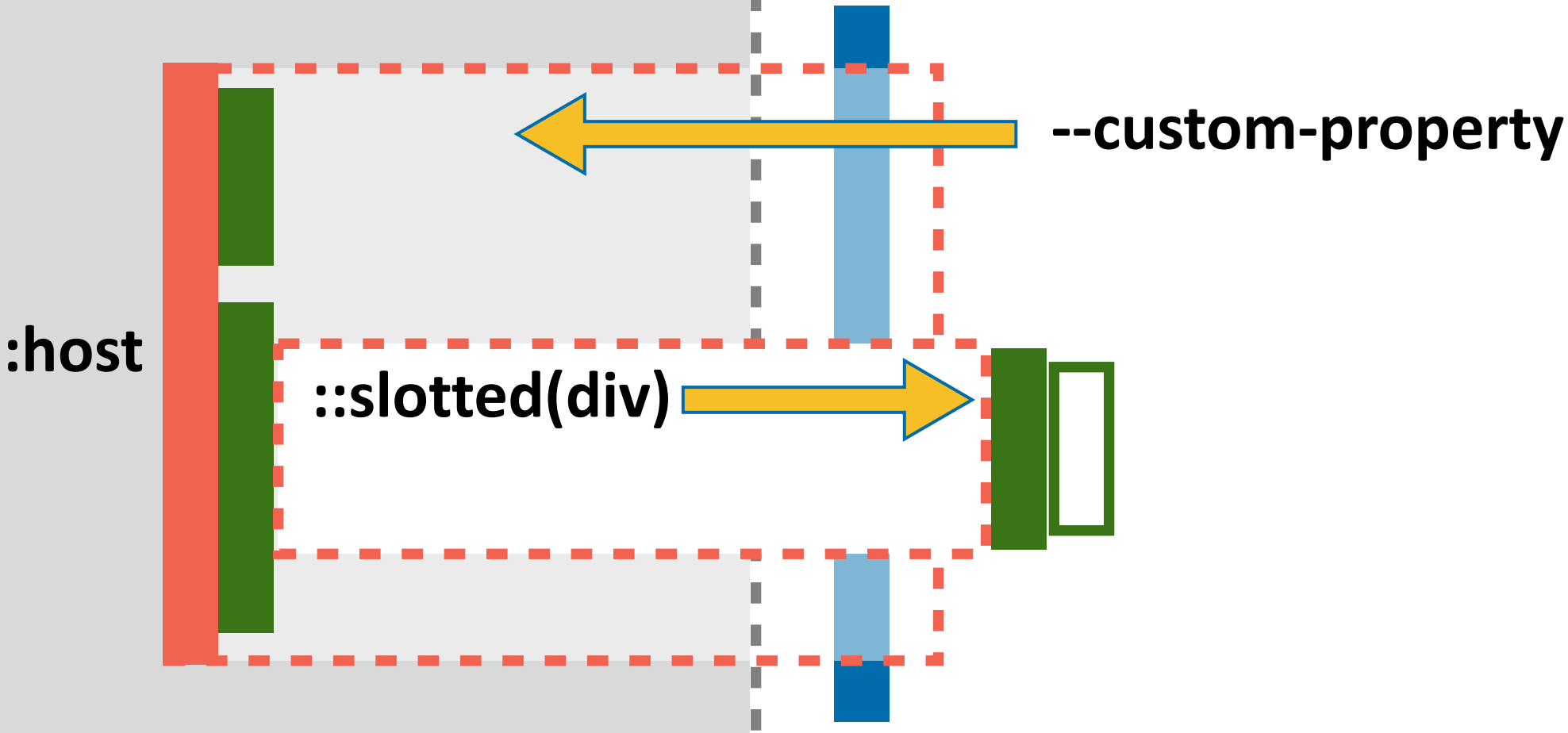
Shadow DOM

Light DOM

Shadow DOM + CSS



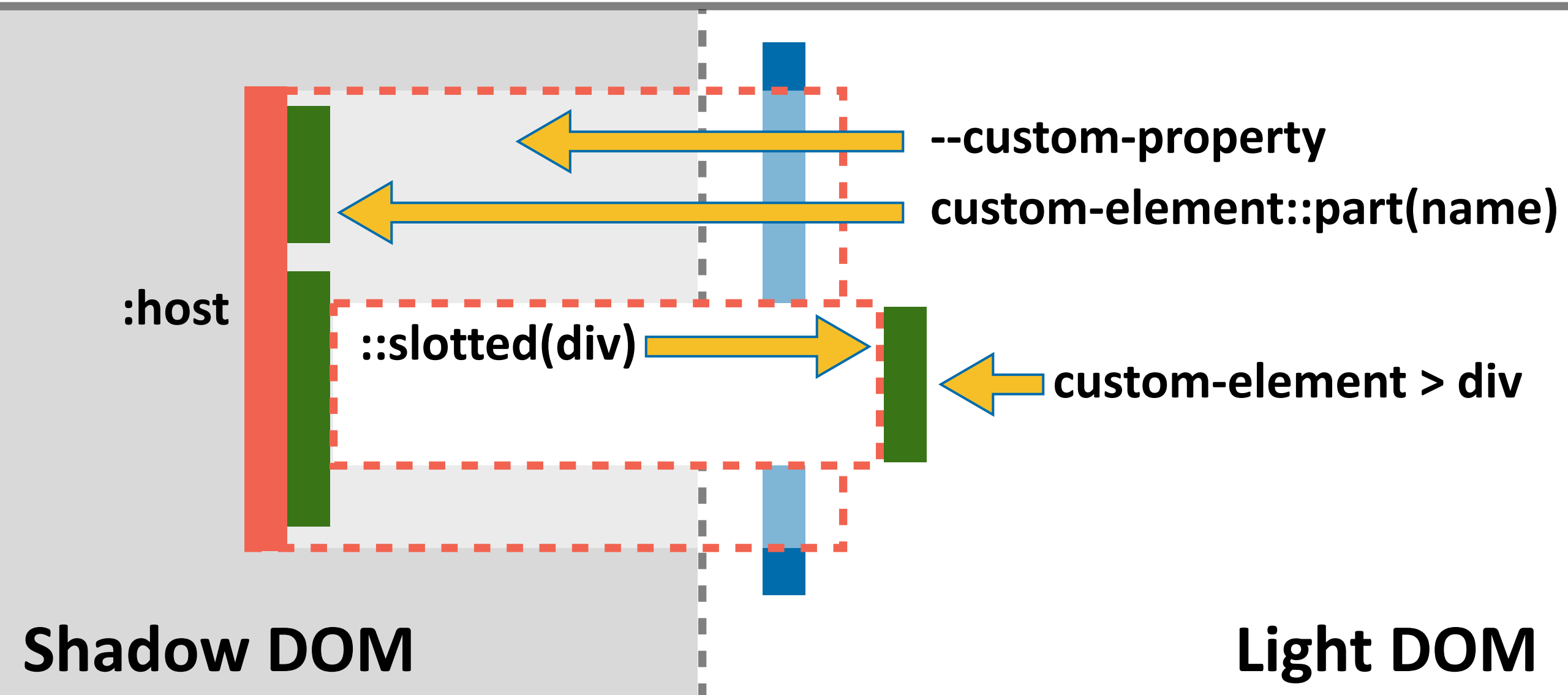
Shadow DOM + CSS



Shadow DOM

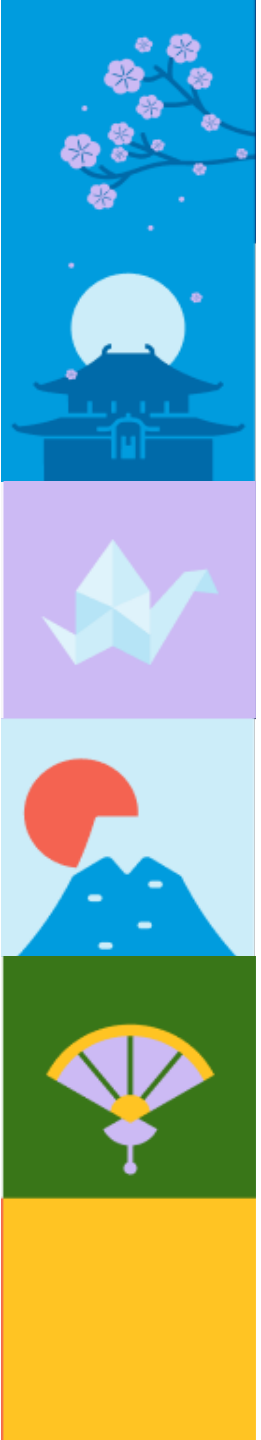
Light DOM

Shadow DOM + CSS



Shadow DOM components

1. Custom elements
2. Templates
3. Shadow DOM



Light DOM components

1. Custom elements
2. ~~Templates~~
3. ~~Shadow DOM~~

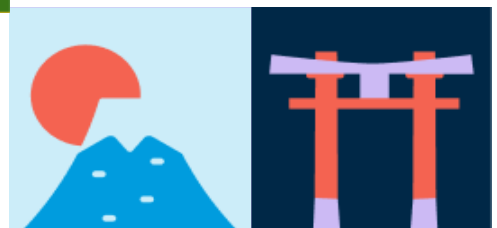
DrupalCon
Nara 2025
17-19 November

Split components

	Shadow DOM Components	Light DOM Components
SERVER	vanilla HTML w/ Declarative Shadow DOM	vanilla HTML w/ custom elements CSS
BROWSER	vanilla JavaScript w/ attachShadow()	vanilla JavaScript w/ customElements.define()

Why choose the light?

1. Less CSS drama:
   Standard CSS Cascade
2. Less accessibility drama:
   ARIA and the Shadow DOM



Single Directory Web Components

using

Light DOM Web Components

or

Declarative Shadow DOM Components

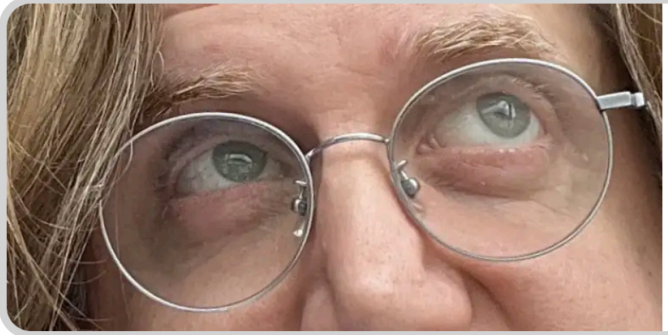
Demo #5: SDC Web Components

An h3 heading

The contents of the card.

FLIP CARD

FLIP



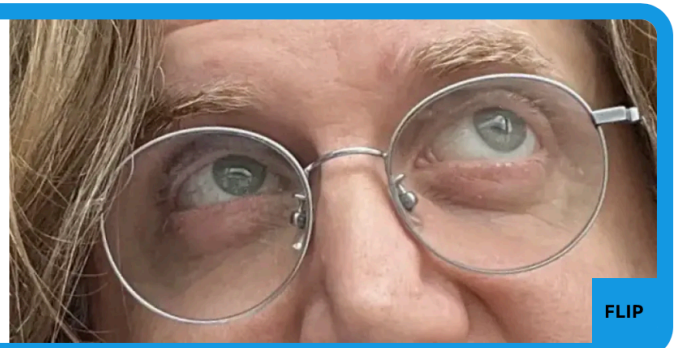
This card doesn't have a heading.

Ask me how!

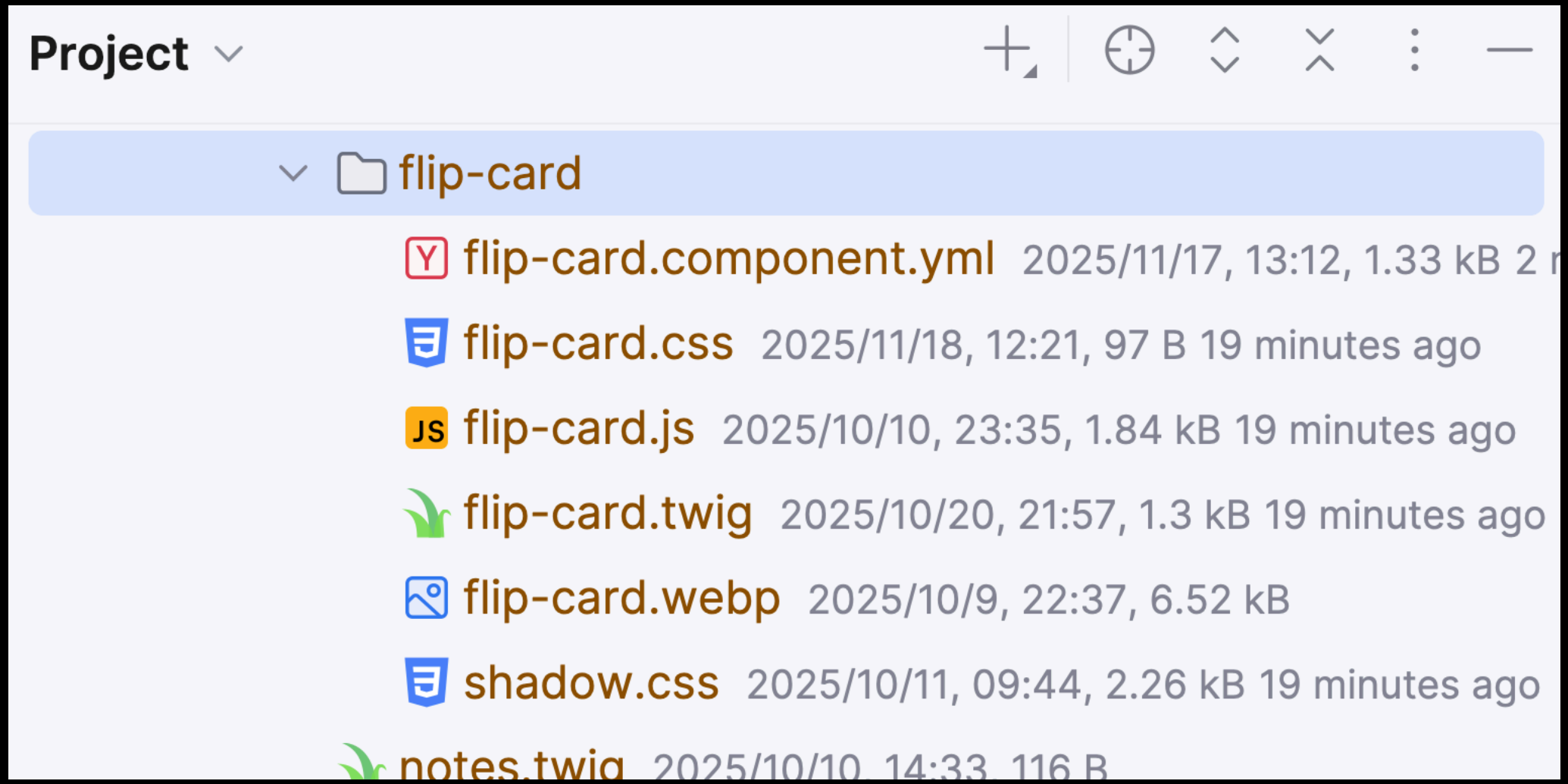
FLOP

Drupal field text

Text from a *CKEditor-enabled* field.



FLIP


Project
flip-card
flip-card.component.yml 2025/11/17, 13:12, 1.33 kB 2 r
flip-card.css 2025/11/18, 12:21, 97 B 19 minutes ago
flip-card.js 2025/10/10, 23:35, 1.84 kB 19 minutes ago
flip-card.twig 2025/10/20, 21:57, 1.3 kB 19 minutes ago
flip-card.webp 2025/10/9, 22:37, 6.52 kB
shadow.css 2025/10/11, 09:44, 2.26 kB 19 minutes ago
notes.twig 2025/10/10, 14:33, 116 B

flip-card.component.yml | flip-card.twig | shadow-root-styles.html.twig | flip-card.js | f

```yaml
name: Flip Card
description: Displays image and text with a flip card style.
status: stable
variants:
  primary:
    title: Primary
    description: Highlight the flip card
props:
  type: object
  properties:
    image_side:
      type: string
      title: Image side
      description: >
        Which side of the card the image should appear on
      enum:
        - left
        - right
      meta:enum:
        left: Left
        right: Right
      x-translation-context: Direction
      default: right
    heading_tag:
```

flip-card.js | flip-card.css | shadow.css | node--5.html.twig

```
{% embed 'demo:flip-card' with {
  variant: 'primary',
  attributes: create_attribute().addClass
   ('using-drupal-fields'),
  image_side: 'right',
  content,
} only %}
  {% block heading %}
    {{ content.field_flip_card_heading }}
  {% endblock %}
  {% block content %}
    {{ content.field_flip_card_text }}
  {% endblock %}
  {% block image %}
    {{ content.field_flip_card_image }}
  {% endblock %}
{% endembed %}
```

```
<flip-card{{ attributes }}>
  <template shadowrootmode="open" shadowrootclonable="true">
    {{ include('@demo/shadow-root-styles.html.twig', {componentMetadata},
     with_context: false) }}

    <div class="text" part="text"...>

    <div class="image" part="image"...>

    <button>{{ (image_side == 'left') ? 'FLOP'|t : 'FLIP'|t }}</button>
  </template>

  {% if block('heading') %}
    <{{ heading_tag|default('h2') }} slot="heading">
      {% block heading %}{% endblock %}
    </{{ heading_tag|default('h2') }}>
  {% endif %}

  {% block content %}
    <p>{{ 'This component requires the "content" slot to be used.'|t }}</p>
  {% endblock %}
```

flip-card.component.yml | flip-card.twig | shadow-root-styles.html.twig | flip-card.js | flip-card.css | sha

```
{# Include with:
 #   {{ include('@demo/shadow-root-styles.html.twig', {componentMetadata}, with_context: false) }}
 #}
<link rel="stylesheet" href="/{{ active_theme_path() }}/css/base/base-for-light-and-shadow-dom.css" />
<link rel="stylesheet" href="/{{ componentMetadata.path }}/shadow.css" />
```

flip-card.component.yml | flip-card.twig | shadow-root-styles.html.twig | flip-card.js | flip-card.css | shadow.css

```twig
{% if image_side == 'left' %}
  {% set attributes = attributes.setAttribute('image', image_side) %}
{% endif %}
{% if variant == 'primary' %}
  {% set attributes = attributes.setAttribute(
    'style',
    '--flip-card-border-color: var(--color--primary-50);
     --flip-card-border-size: 10px',
  ) %}
{% endif %}

<flip-card{{ attributes }}>
  <template shadowrootmode="open" shadowrootclonable="true">
    {{ include('@demo/shadow-root-styles.html.twig', {componentMetadata},
     with_context: false) }}

    <div class="text" part="text"...>

    <div class="image" part="image"...>
```

```
25
26        <button>{{ (image_side == 'left') ? 'FLOP'|t : 'FLIP'|t }}</button>
27      </template>
28
29      {% if block('heading') %}
30        <{{ heading_tag|default('h2') }} slot="heading">
31          {% block heading %}{% endblock %}
32        </{{ heading_tag|default('h2') }}>
33      {% endif %}
34
35      {% block content %}
36        <p>{{ 'This component requires the "content" slot to be used.'|t }}</p>
37      {% endblock %}
38
39      <div slot="image">
40        {% block image %}
41          <img alt="{{ 'Flip card'|t }}" src="/{{ componentMetadata.path
            }}/flip-card.webp" />
42        {% endblock %}
43      </div>
44    </flip-card>
```

flip-card.component.yml | flip-card.twig | shadow-root-styles.html.twig | flip-card.js | flip-card.css | sh

```
1   class FlipCard extends HTMLElement {
14    constructor() {
15      super();
16
17      // Attach Shadow DOM if there isn't already a declarative one.
18      if (!this.shadowRoot) {
19        this.attachShadow( init: {
20          mode: 'open',
21          clonable: true,
22        });
23      }
24    }
25

      Show usages
26    connectedCallback() : void  {
27      // Grab the template inside our custom element.
28      const template : HTMLTemplateElement  = this.querySelector
          ( selectors: 'template');
29
30      // Clone the template into the Shadow. If the browser supports declarative
31      // Shadow DOM, the tempate will no longer be present as a Light DOM child.
32      if (template) {
33        this.shadowRoot.appendChild(template.content.cloneNode( deep: true));
34      }
```

flip-card.component.yml | flip-card.twig | shadow-root-styles.html.twig | flip-card.js | flip-card.css

```
1  /* Loaded by SDC into the Light DOM */
2
3  flip-card::part(text) {
4    background-color: lightpink;
5  }
```


Project
flip-card
flip-card.component.yml
flip-card.css 2025/11/18, 12
flip-card.js 2025/10/10, 23:
flip-card.twig 2025/10/20,
flip-card.webp 2025/10/9,
notes.twig 2025/10/10, 14:3
shadow.css 2025/10/11, 09
shadow-dom
flip-card.component.yml
flip-card.twig
shadow-root-styles.html.twig
flip-card.js
flip-card.css
shadow.css
5
12
:host {
--padding: var(--flip-card-padding, 20px);
--radius: var(--flip-card-radius, var(--radius-default, 20px));
--border-color: var(--flip-card-border-color, lightgrey);
--border-size: var(--flip-card-border-size, 5px);
position: relative;

THANK YOU!

DrupalCon
Nara 2025
17-19 November

QUESTIONS?

Drupal: JohnAlbin

email: john@albin.net

https:// **john.albin.net/presentations**