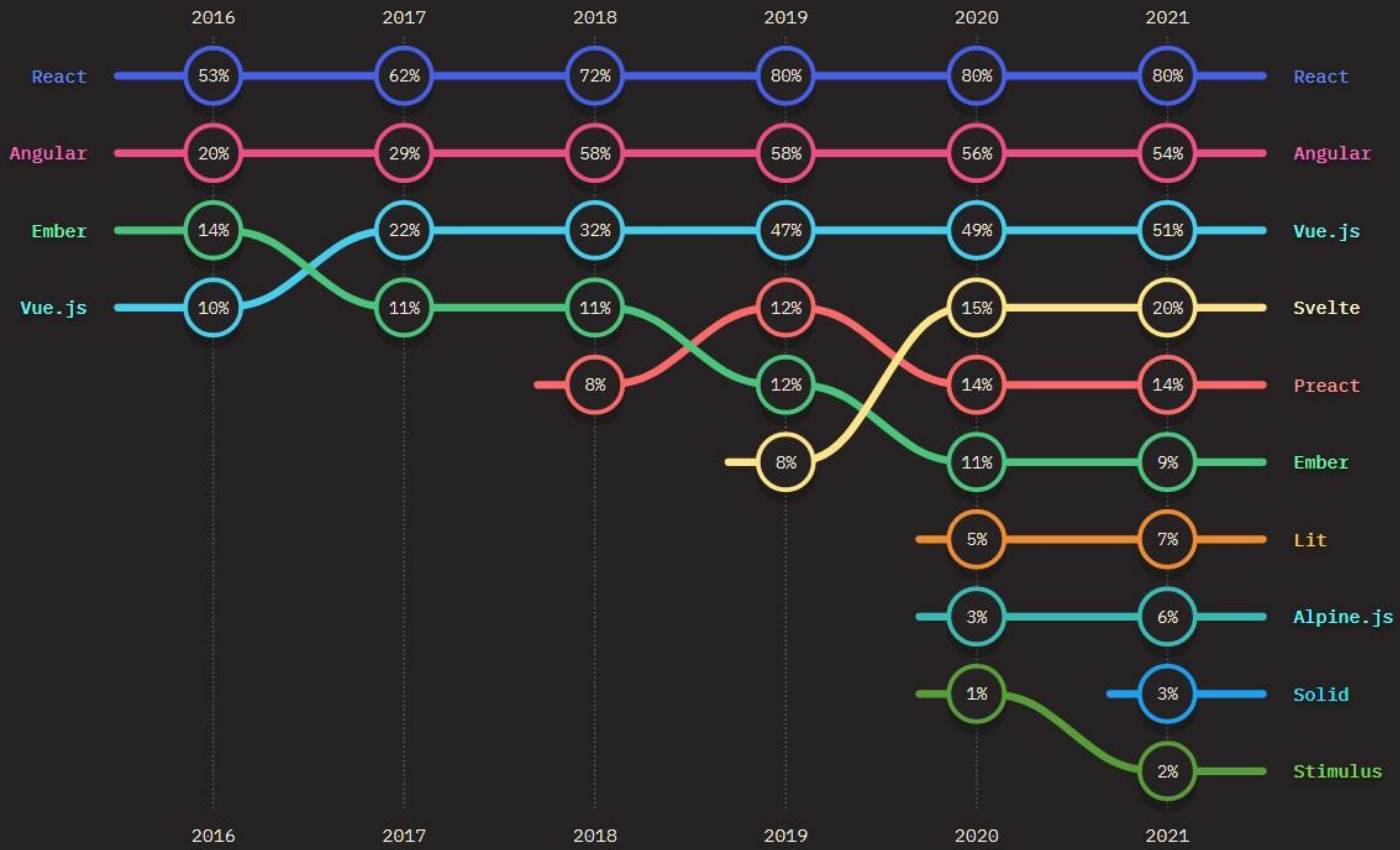
cognizant softvision

PROGRAMMERS' WEEK 2022

Develop standard web components

Under a design system


2016
2017
2018
2019
2020
2021
React
Angular
Ember
Vue.js
53%
62%
72%
80%
80%
80%
20%
29%
58%
58%
56%
54%
14%
22%
32%
47%
49%
51%
10%
11%
11%
12%
15%
20%
8%
12%
14%
14%
8%
11%
9%
5%
7%
3%
6%
1%
3%
2%
React
Angular
Vue.js
Svelte
Preact
Ember
Lit
Alpine.js
Solid
Stimulus

Gustavo Petruzzi

PROGRAMMERS´ WEEK 2022

Frontend developer

Work with Angular and React.

Love to ride my motorcycle and learn about different technologies.

PROGRAMMERS´
WEEK 2022

Agenda

Design system

Web components

Web components in Angular

Web components in React

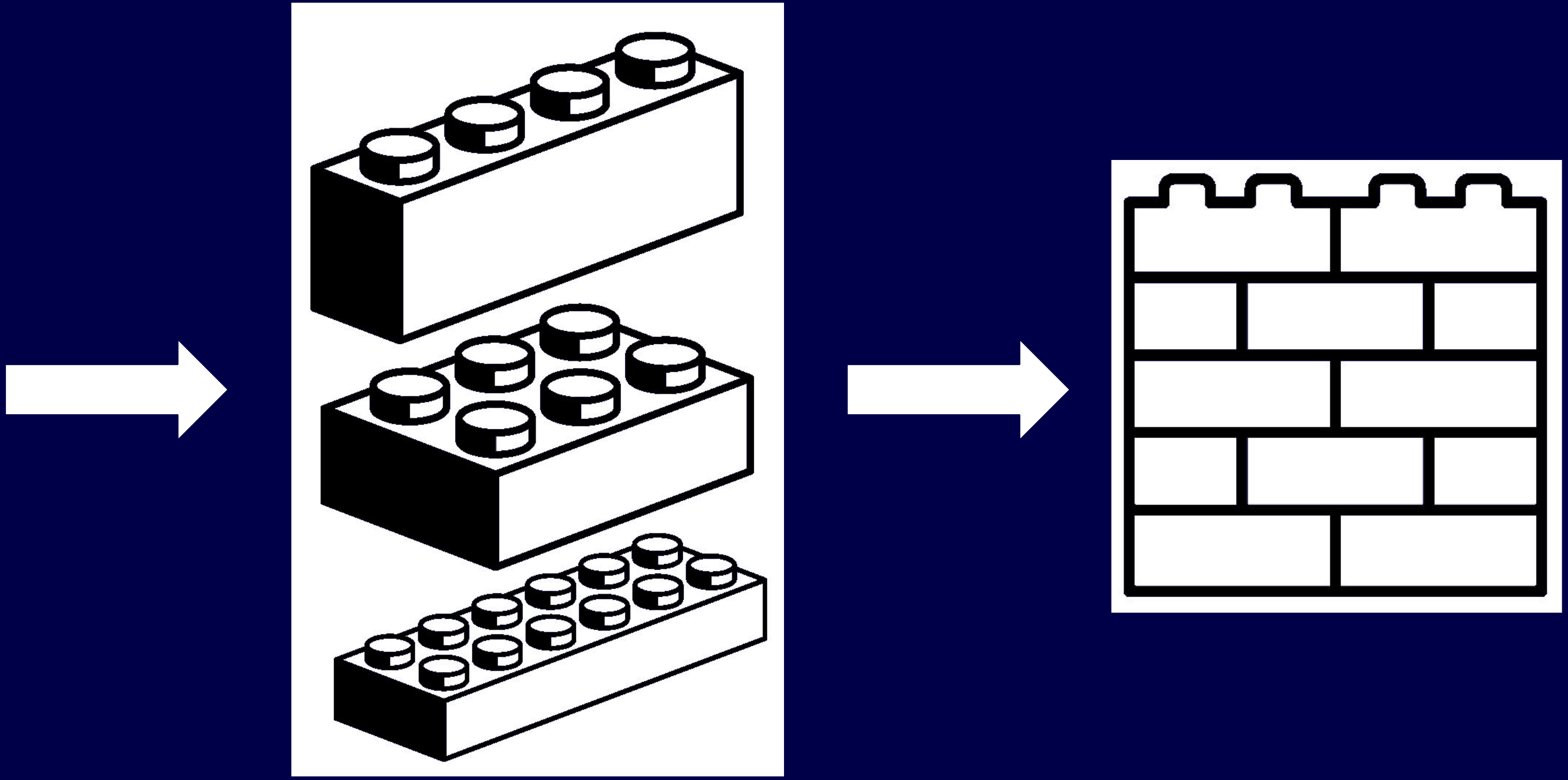
Final Thoughts

Q & A

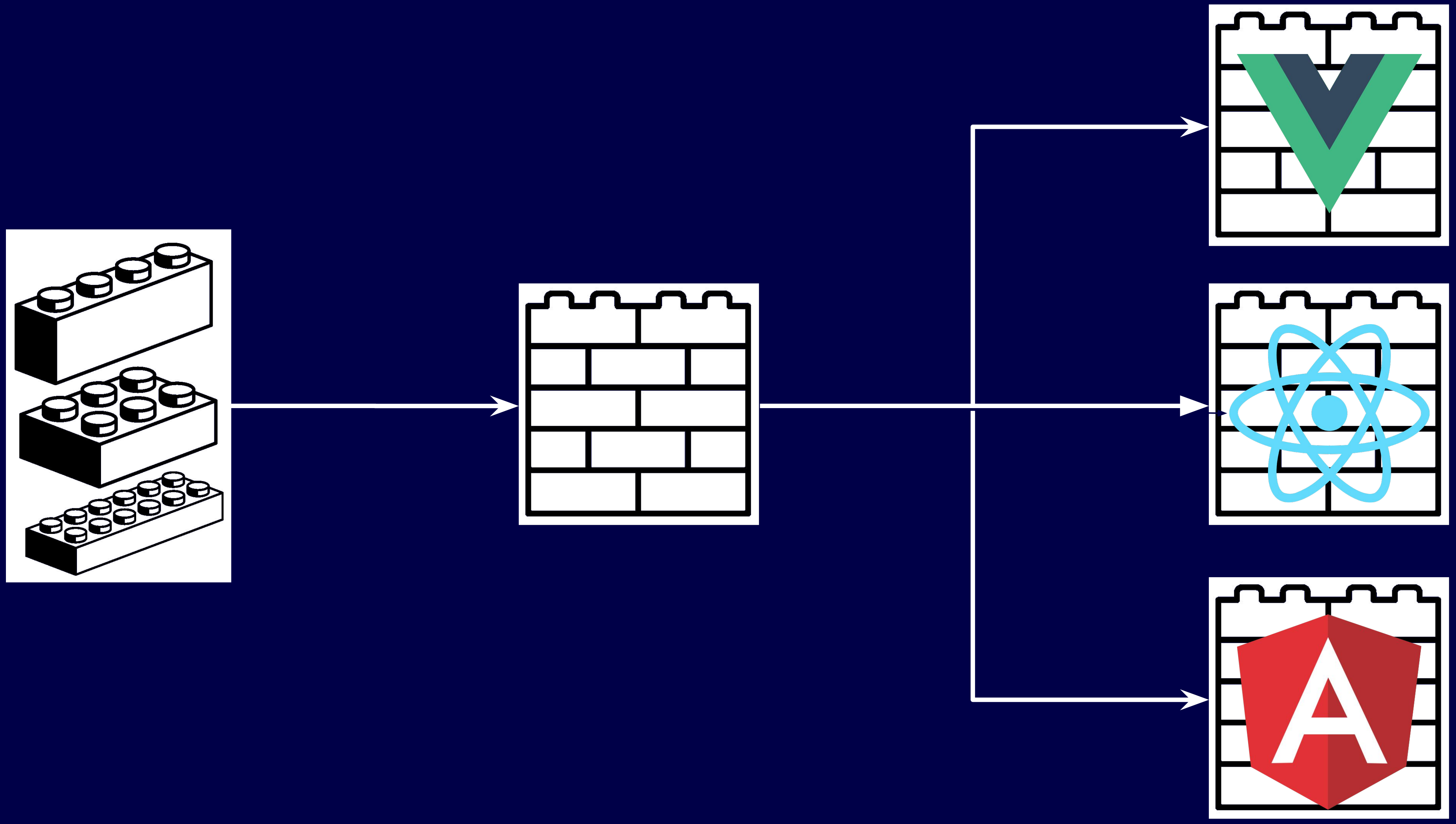
cognizant
softvision

What is a design system?

Design system



The problem with design system and frameworks.



Working Sample
React & Angular

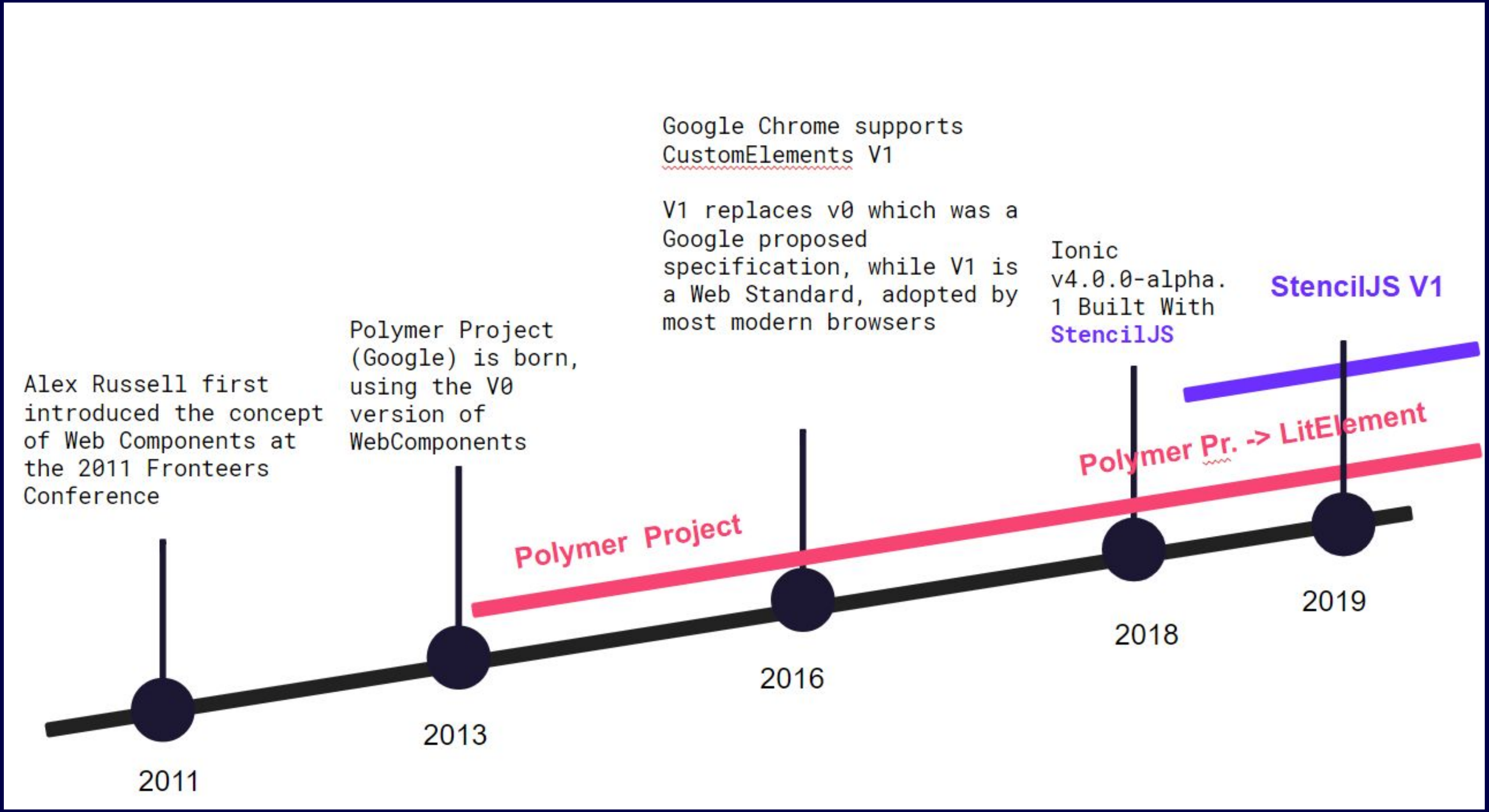
Web components

What are web components?

Web components is a suite of different technologies that allows you create reusable and encapsulated components.

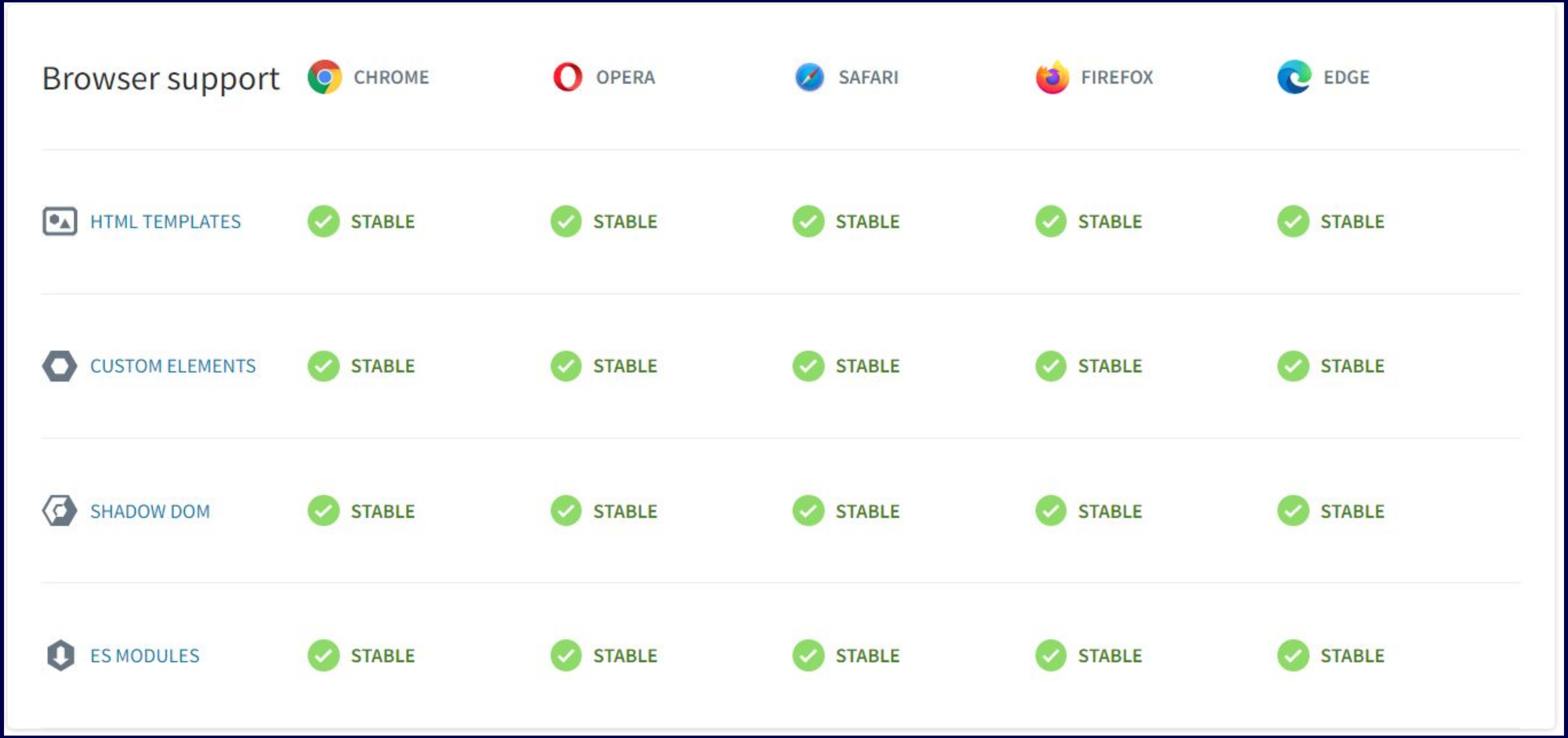
To create web components, you only need Javascript, HTML and CSS.

Web components first appearance


Alex Russell first introduced the concept of Web Components at the 2011 Fronteers Conference
Polymer Project (Google) is born, using the V0 version of WebComponents
Google Chrome supports CustomElements V1
V1 replaces v0 which was a Google proposed specification, while V1 is a Web Standard, adopted by most modern browsers
Ionic v4.0.0-alpha.1 Built With StencilJS
StencilJS V1
Polymer Project
Polymer Pr. -> LitElement
2011
2013
2016
2018
2019

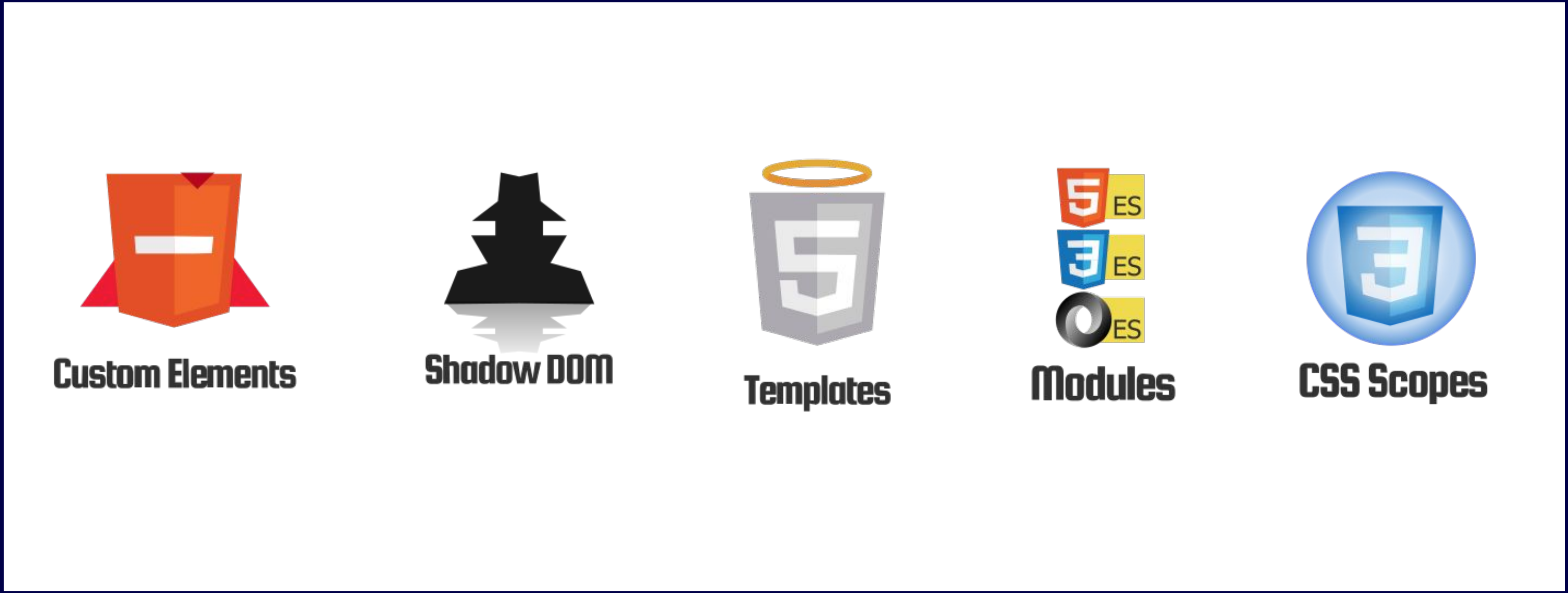
Web components adoption

Browser support



Browser support	CHROME	OPERA	SAFARI	FIREFOX	EDGE
HTML TEMPLATES	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE
CUSTOM ELEMENTS	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE
SHADOW DOM	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE
ES MODULES	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE	✓ STABLE

Web component specifications.

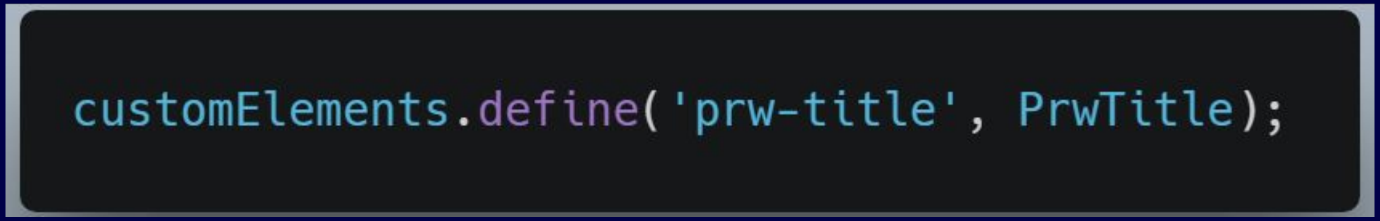


What are custom elements?

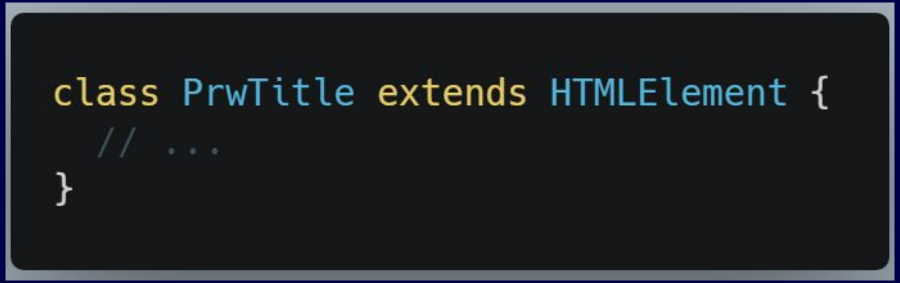
Autonomous custom elements

Customized built-in elements

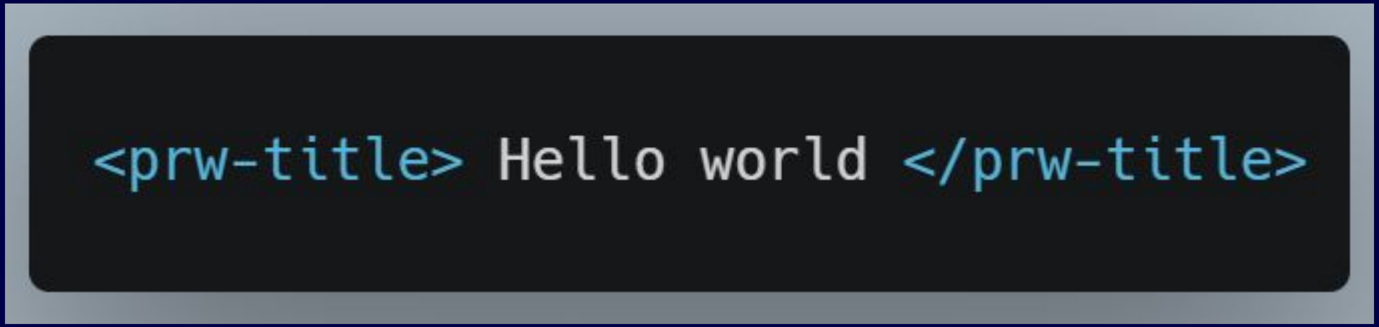
Autonomous custom elements



```
customElements.define('prw-title', PrwTitle);
```

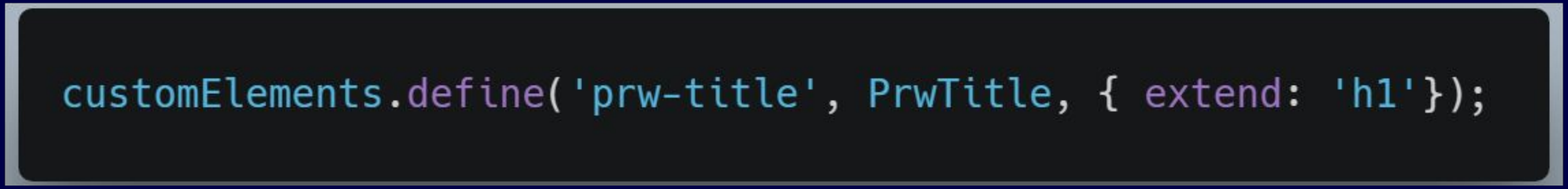


```
class PrwTitle extends HTMLElement {
  // ...
}
```

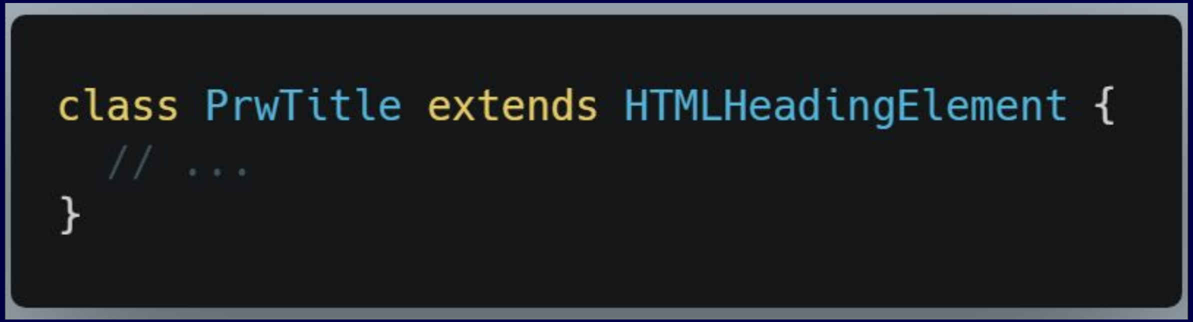


```
<prw-title> Hello world </prw-title>
```

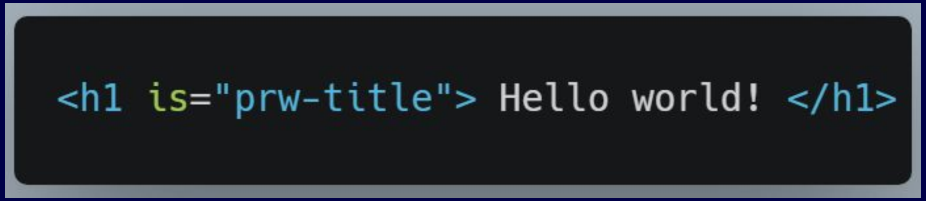
Customized built-in elements



```
customElements.define('prw-title', PrwTitle, { extend: 'h1'});
```



```
class PrwTitle extends HTMLHeadingElement {
  // ...
}
```



```
<h1 is="prw-title"> Hello world! </h1>
```

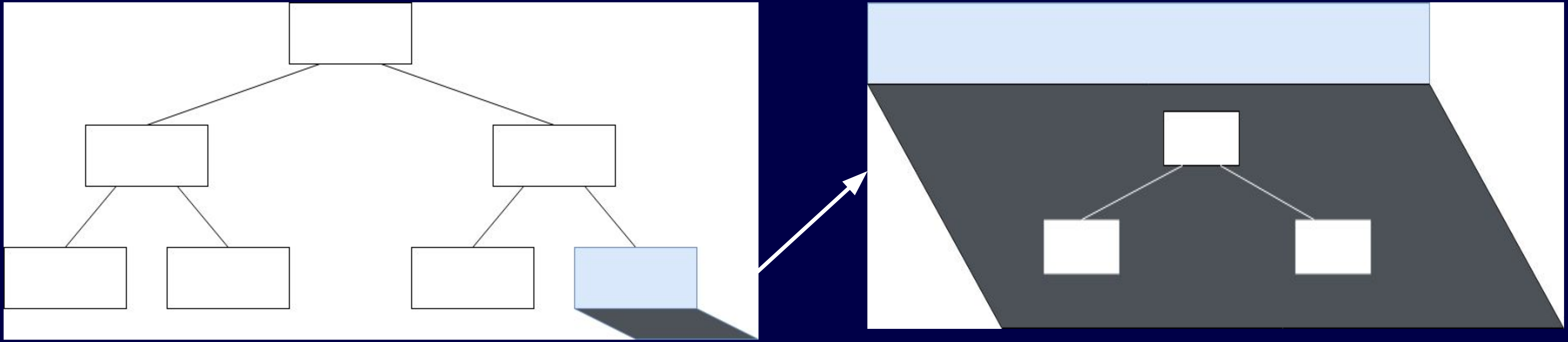
Make sure you do not forget to **define** your web component.

The **class definition is not enough** to create your component.

Browsy's tips

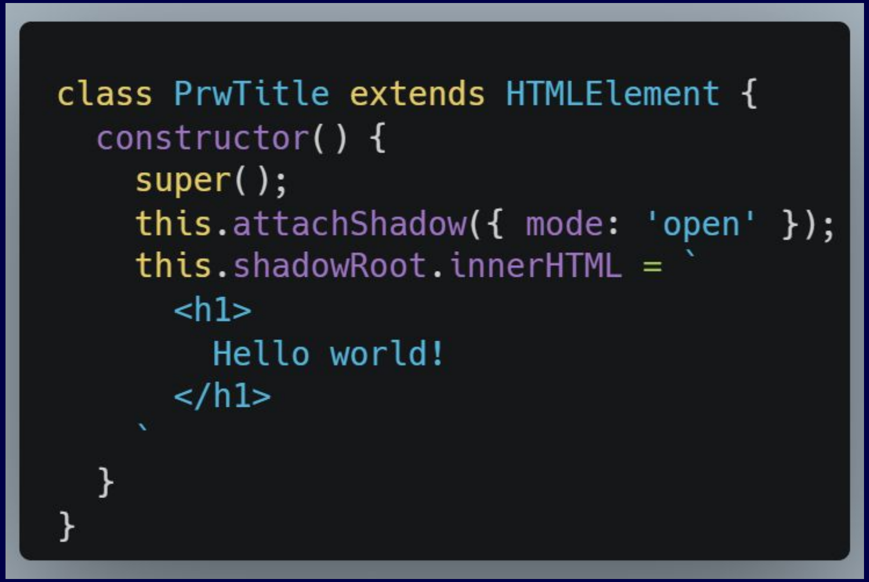
What is the shadow DOM?

Shadow DOM

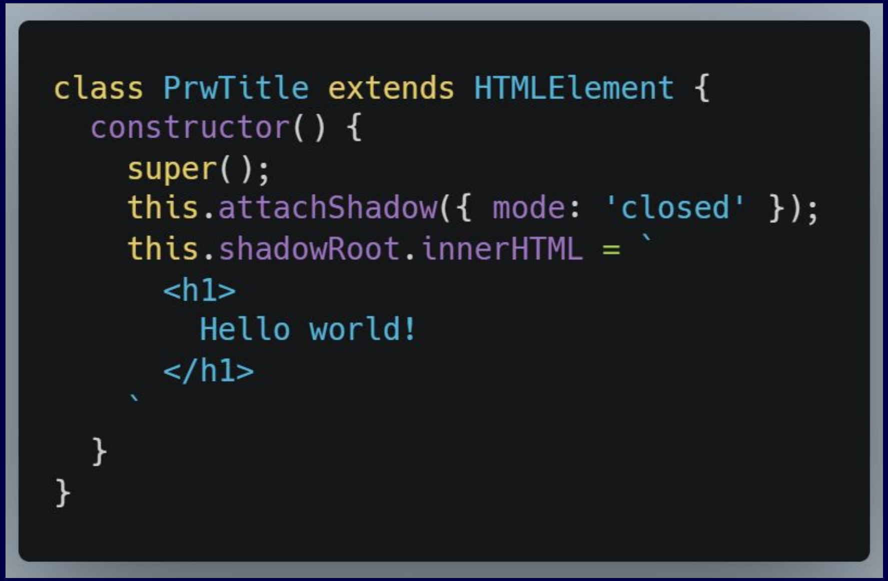


PROGRAMMERS'
WEEK 2022

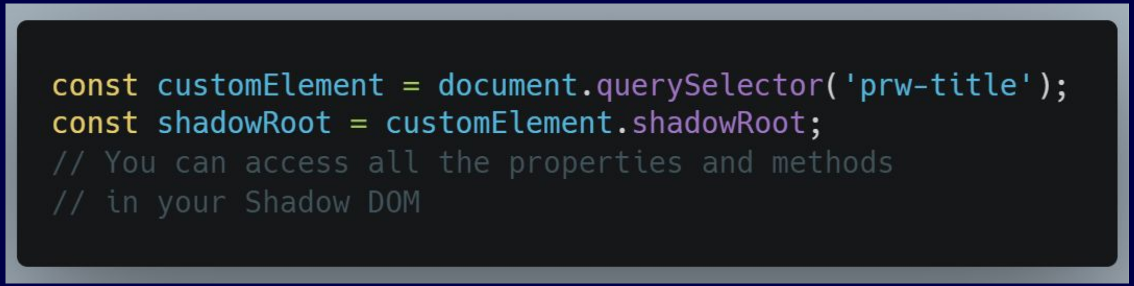
Mode option



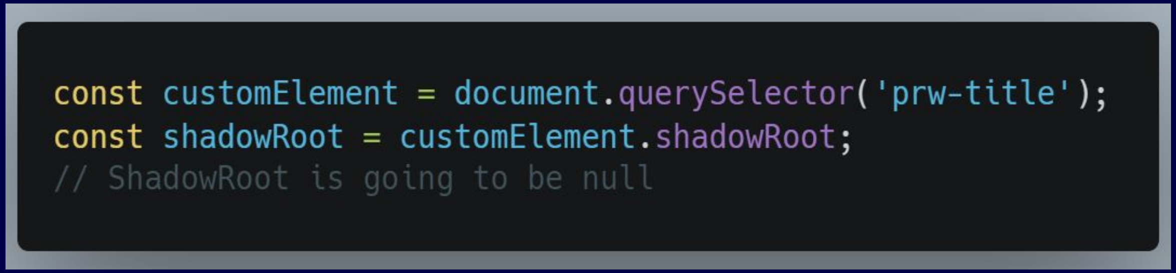
```
class PrwTitle extends HTMLElement {
  constructor() {
    super();
    this.attachShadow({ mode: 'open' });
    this.shadowRoot.innerHTML = `
      <h1>
        Hello world!
      </h1>
    `
  }
}
```



```
class PrwTitle extends HTMLElement {
  constructor() {
    super();
    this.attachShadow({ mode: 'closed' });
    this.shadowRoot.innerHTML = `
      <h1>
        Hello world!
      </h1>
    `
  }
}
```



```
const customElement = document.querySelector('prw-title');
const shadowRoot = customElement.shadowRoot;
// You can access all the properties and methods
// in your Shadow DOM
```



```
const customElement = document.querySelector('prw-title');
const shadowRoot = customElement.shadowRoot;
// ShadowRoot is going to be null
```

cognizant
softvision

Keep in mind that the shadow DOM is not a security measure. The user can anyway affect the shadow DOM of your component, even just by using the attachShadow() method of your shadow host element

Browsy's tips

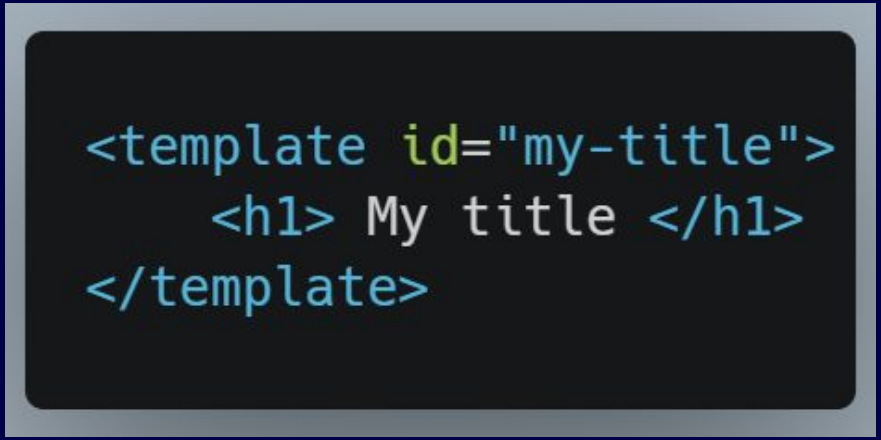
What are templates?

Template tag

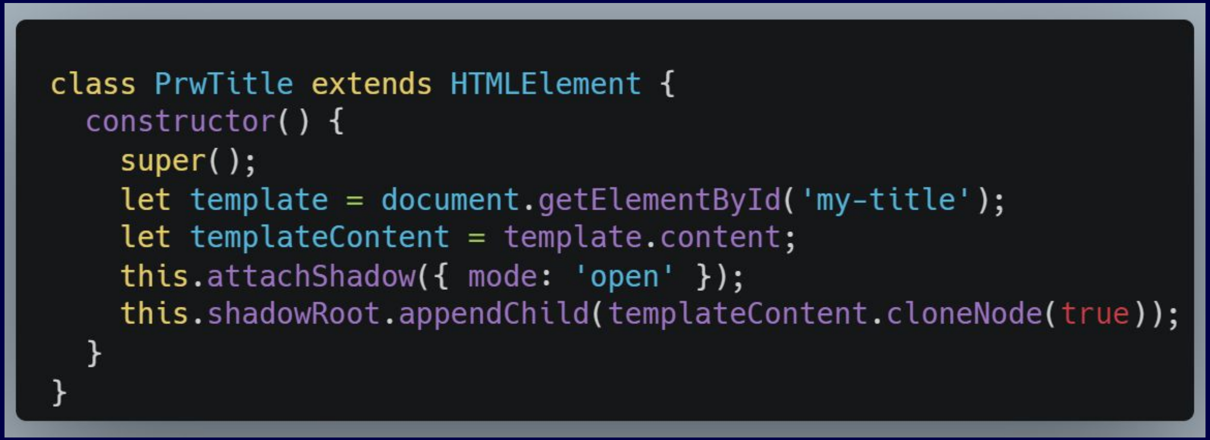
Templates

Slot tag

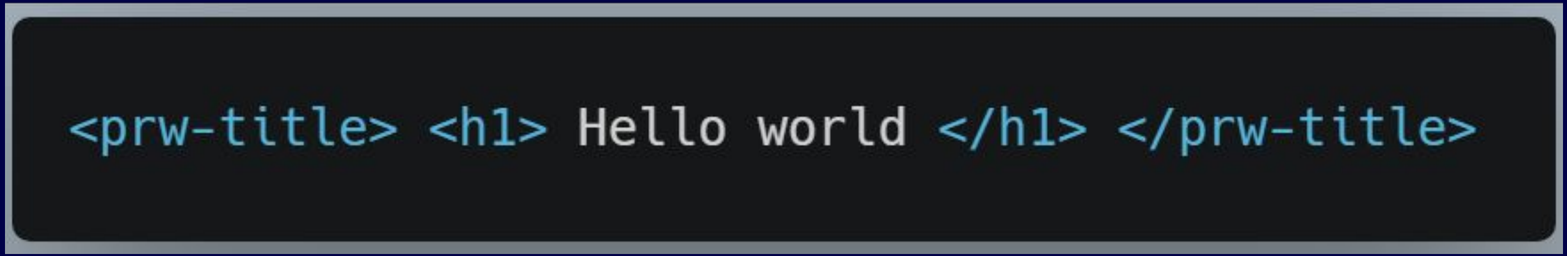
Template tag.



```html
<template id="my-title">
    <h1> My title </h1>
</template>
```

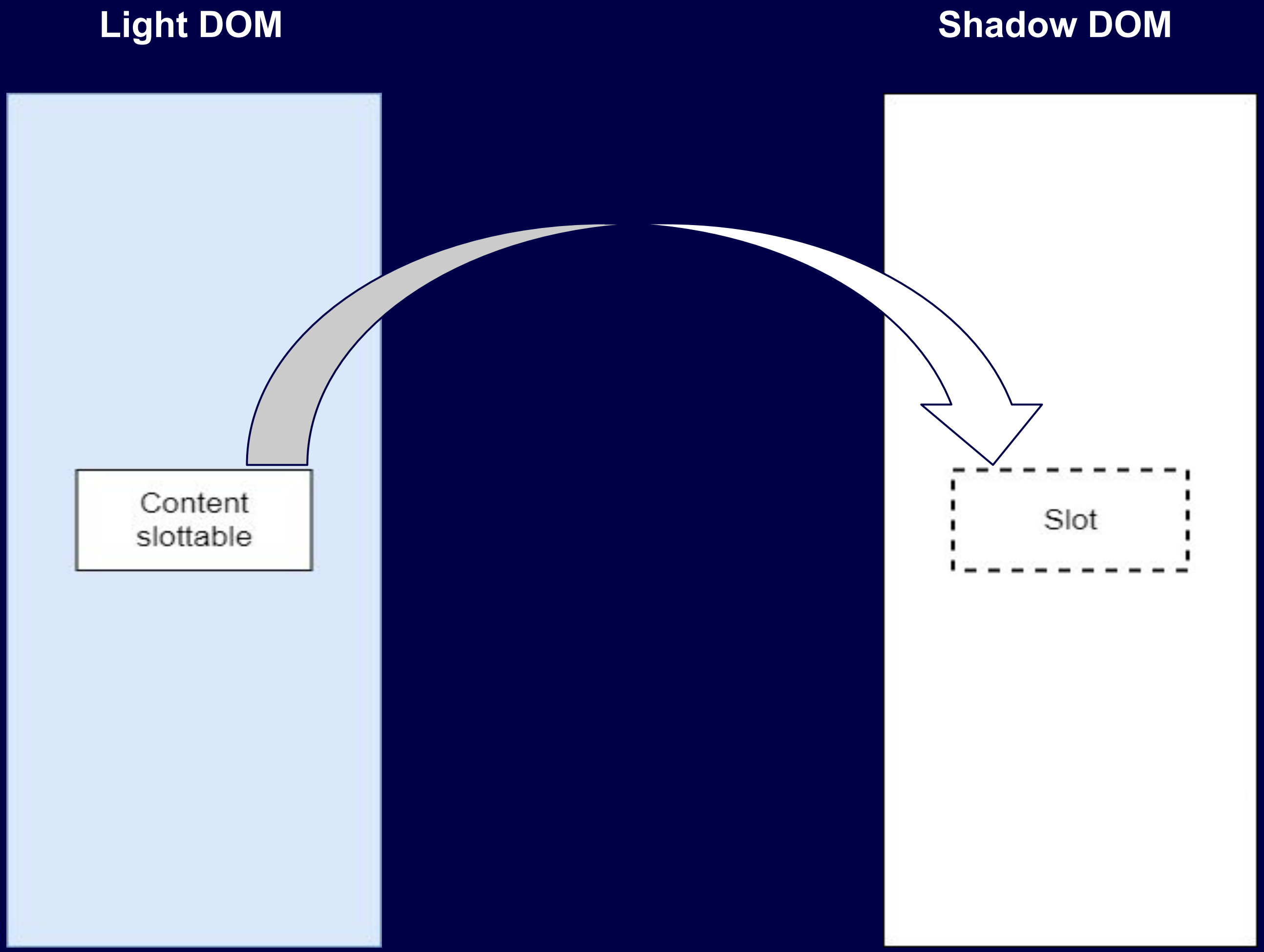


```javascript
class PrwTitle extends HTMLElement {
  constructor() {
    super();
    let template = document.getElementById('my-title');
    let templateContent = template.content;
    this.attachShadow({ mode: 'open' });
    this.shadowRoot.appendChild(templateContent.cloneNode(true));
  }
}
```

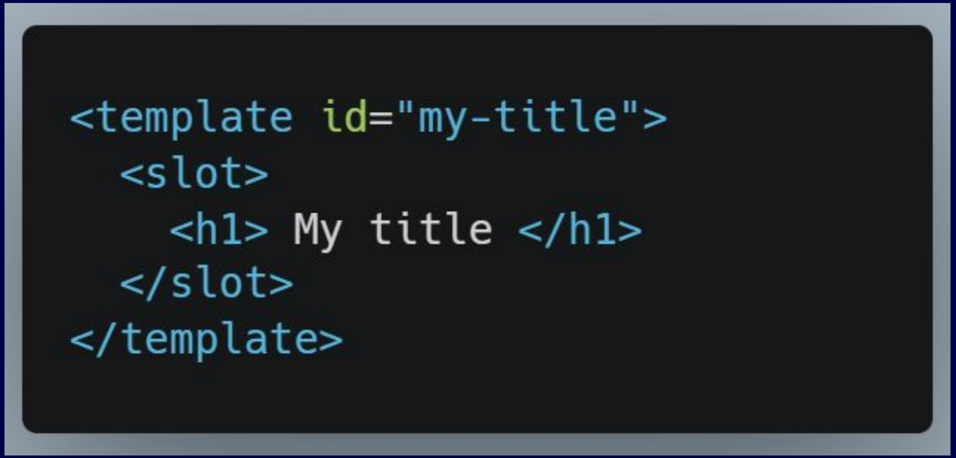


```html
<prw-title> <h1> Hello world </h1> </prw-title>
```

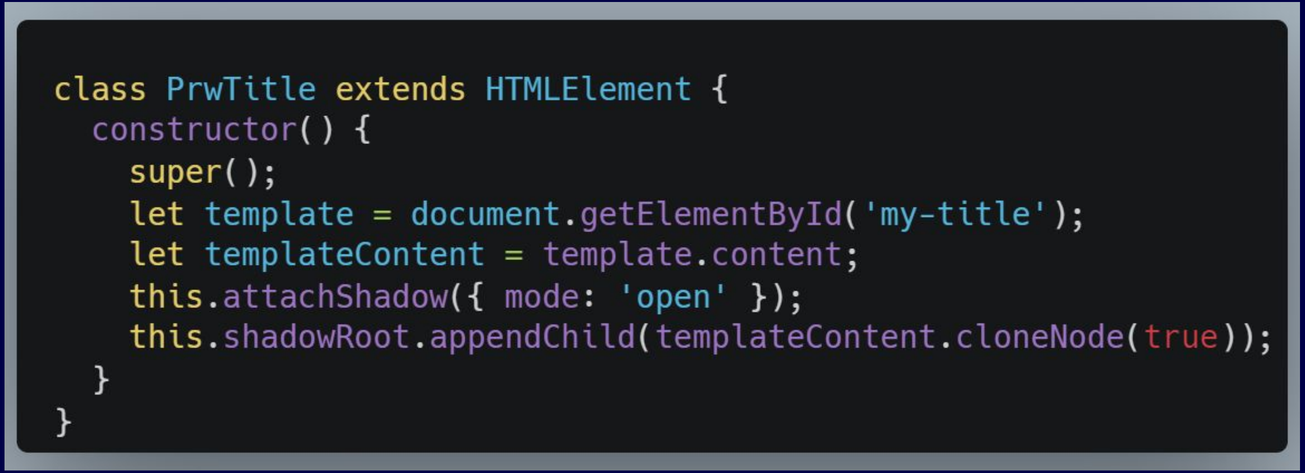
Slot tag.



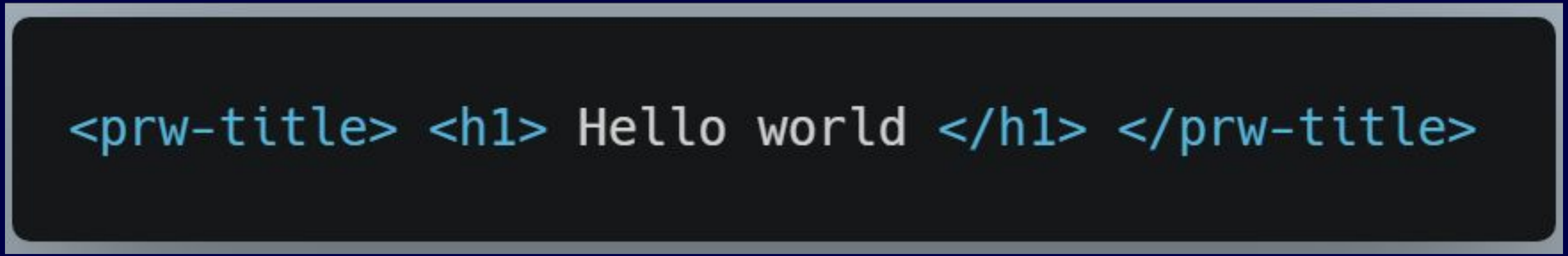
Slot tag.



```html
<template id="my-title">
  <slot>
    <h1> My title </h1>
  </slot>
</template>
```

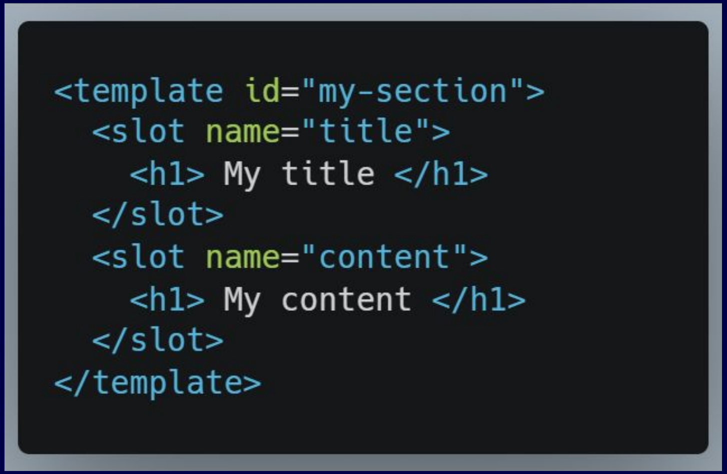


```js
class PrwTitle extends HTMLElement {
  constructor() {
    super();
    let template = document.getElementById('my-title');
    let templateContent = template.content;
    this.attachShadow({ mode: 'open' });
    this.shadowRoot.appendChild(templateContent.cloneNode(true));
  }
}
```

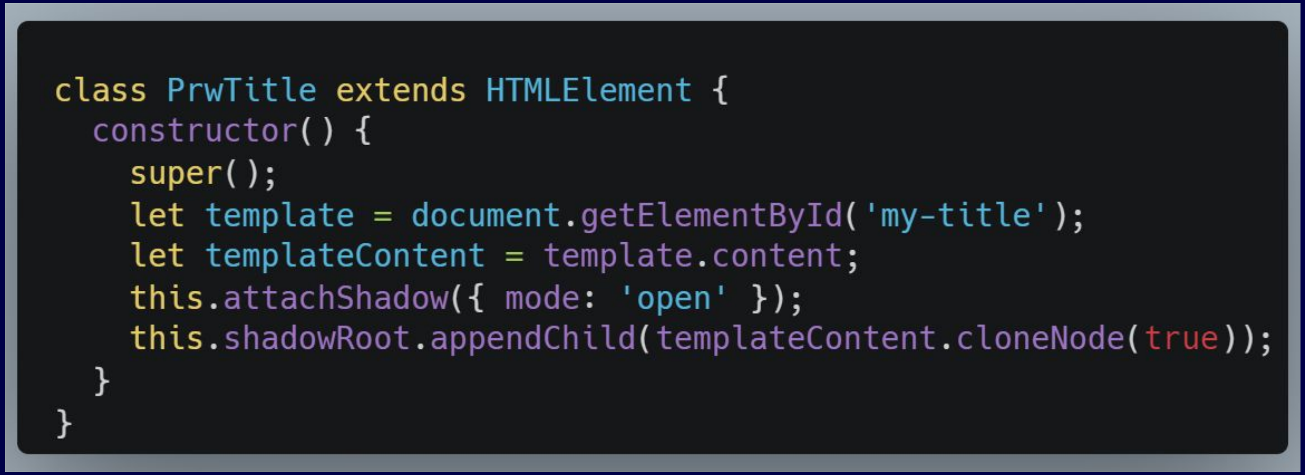


```html
<prw-title> <h1> Hello world </h1> </prw-title>
```

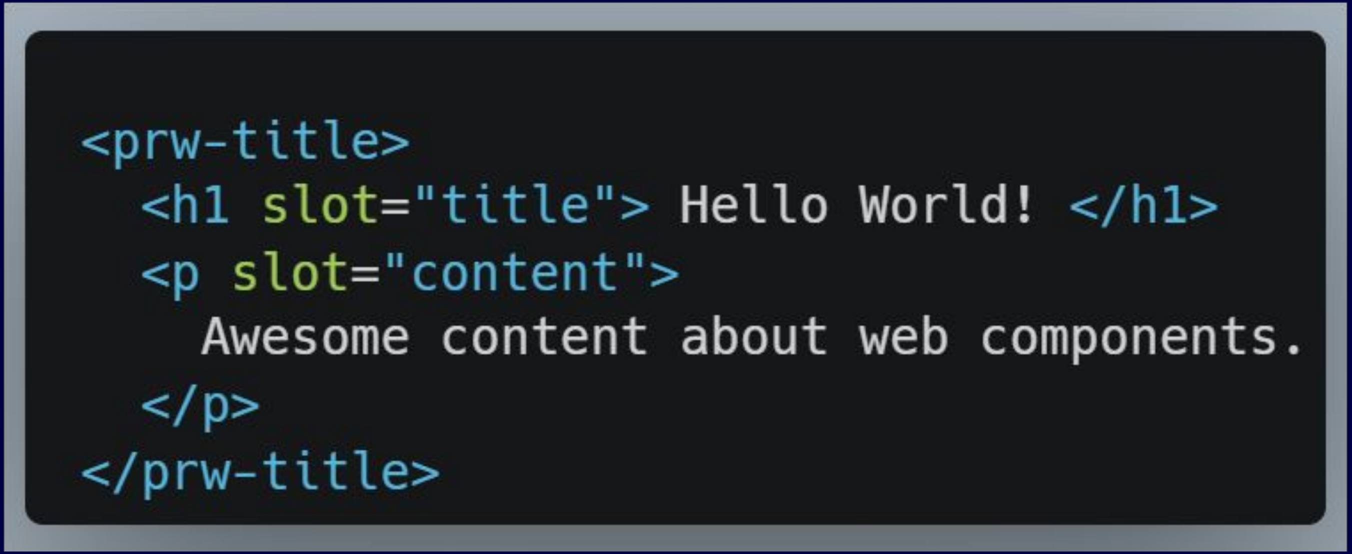
Multiples slots



```
<template id="my-section">
  <slot name="title">
    <h1> My title </h1>
  </slot>
  <slot name="content">
    <h1> My content </h1>
  </slot>
</template>
```

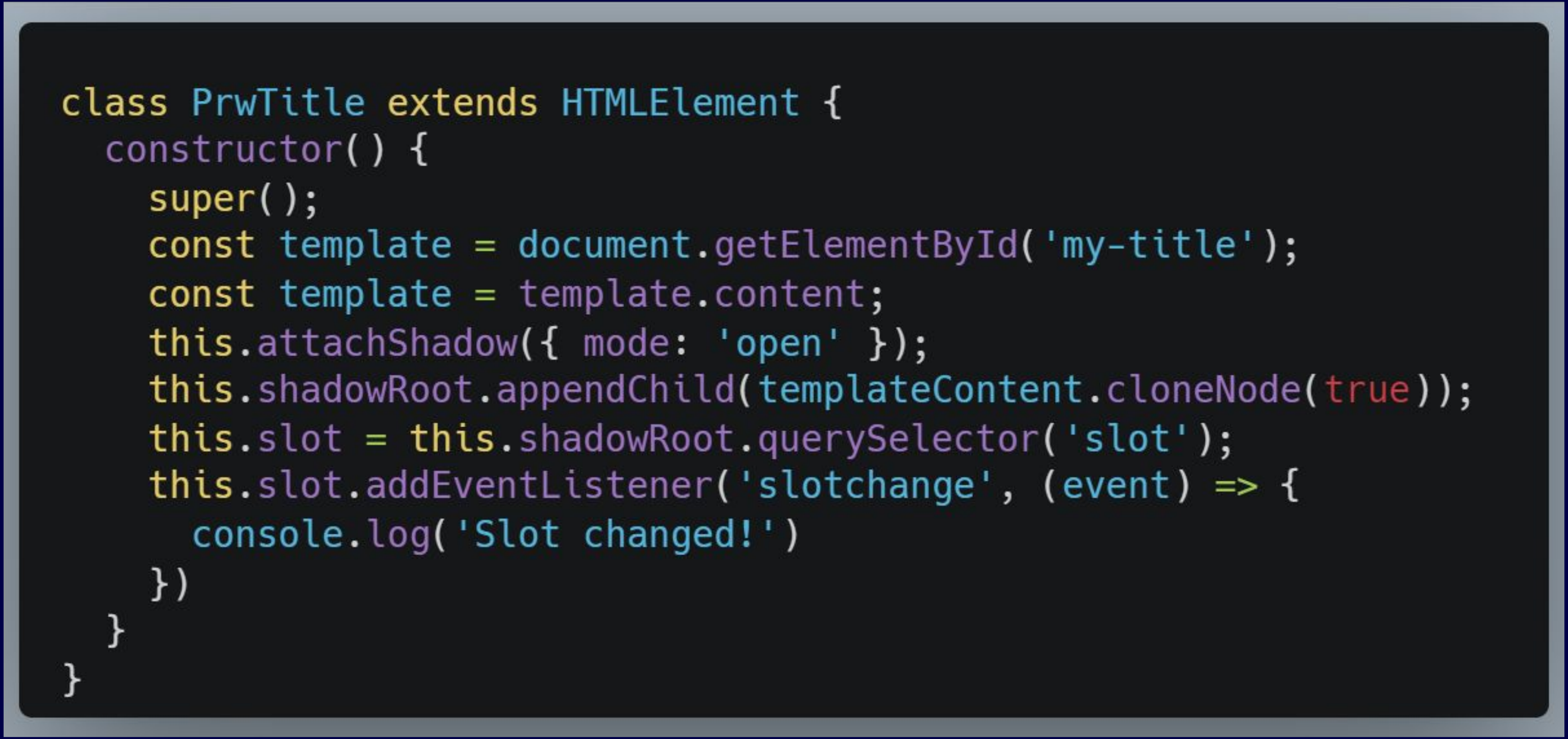


```
class PrwTitle extends HTMLElement {
  constructor() {
    super();
    let template = document.getElementById('my-title');
    let templateContent = template.content;
    this.attachShadow({ mode: 'open' });
    this.shadowRoot.appendChild(templateContent.cloneNode(true));
  }
}
```



```
<prw-title>
  <h1 slot="title"> Hello World! </h1>
  <p slot="content">
    Awesome content about web components.
  </p>
</prw-title>
```

Slotchange event



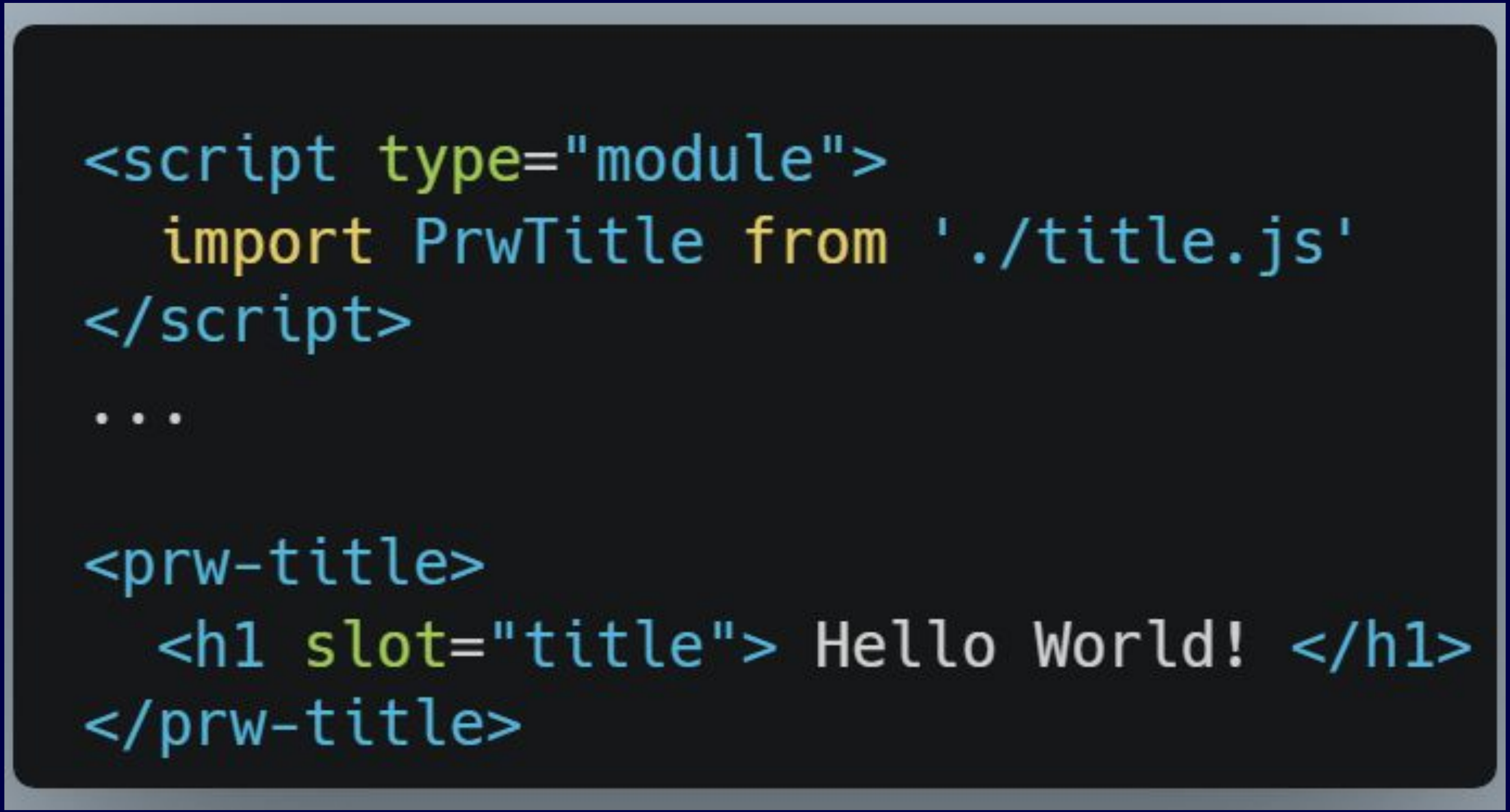
```
class PrwTitle extends HTMLElement {
  constructor() {
    super();
    const template = document.getElementById('my-title');
    const template = template.content;
    this.attachShadow({ mode: 'open' });
    this.shadowRoot.appendChild(templateContent.cloneNode(true));
    this.slot = this.shadowRoot.querySelector('slot');
    this.slot.addEventListener('slotchange', (event) => {
      console.log('Slot changed!')
    })
  }
}
```

Remember! You need a shadow DOM if you want to use slots. Slots are only placeholders inside a shadow DOM.

Browsy's tips

ES modules

ES modules

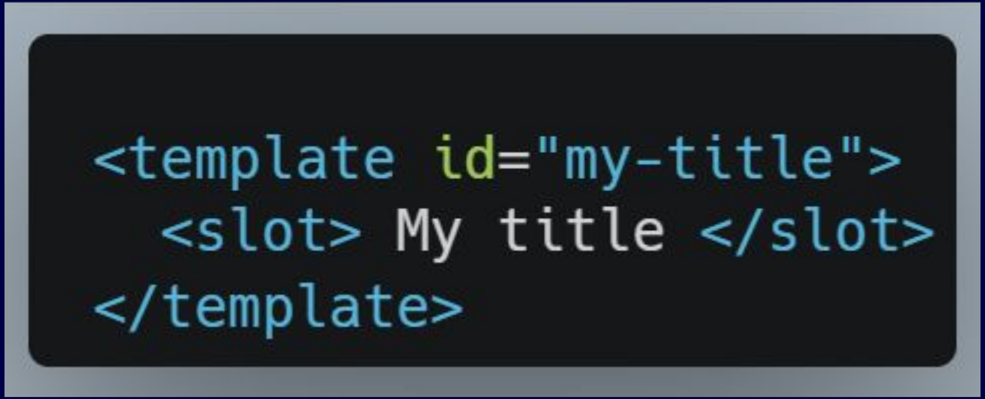


```
<script type="module">
  import PrwTitle from './title.js'
</script>
...

<prw-title>
  <h1 slot="title"> Hello World! </h1>
</prw-title>
```

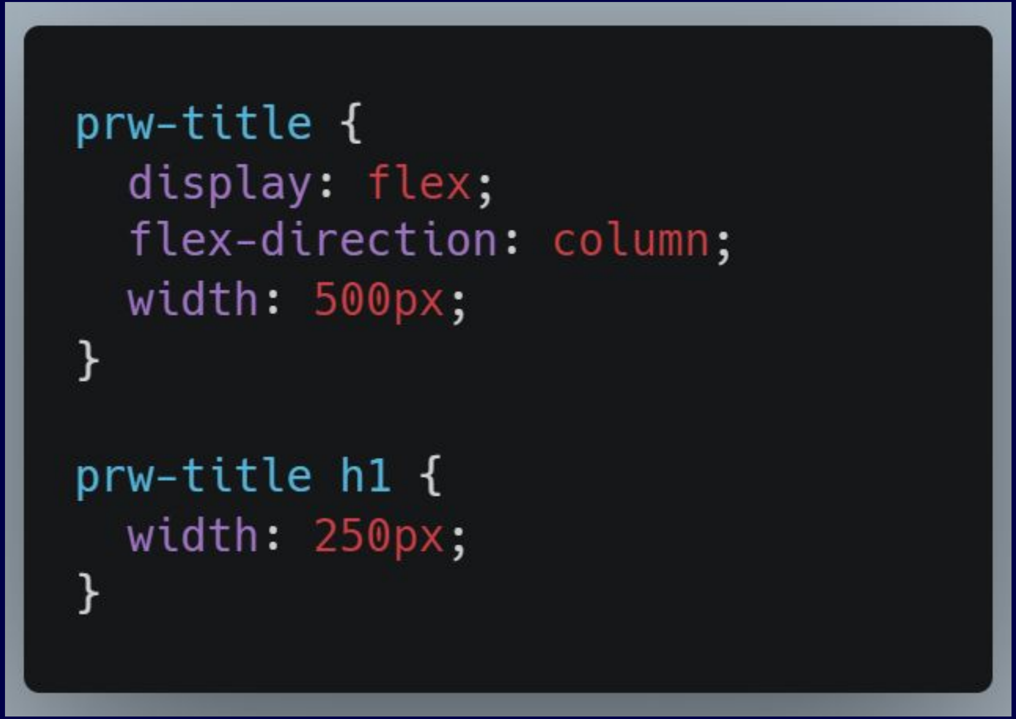
CSS Scopes

PROGRAMMERS´
WEEK 2022

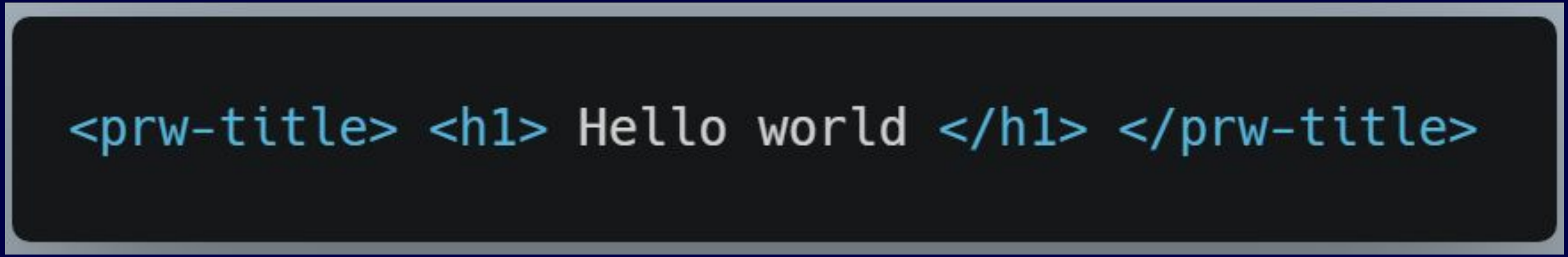
Web component tag



```html
<template id="my-title">
  <slot> My title </slot>
</template>
```



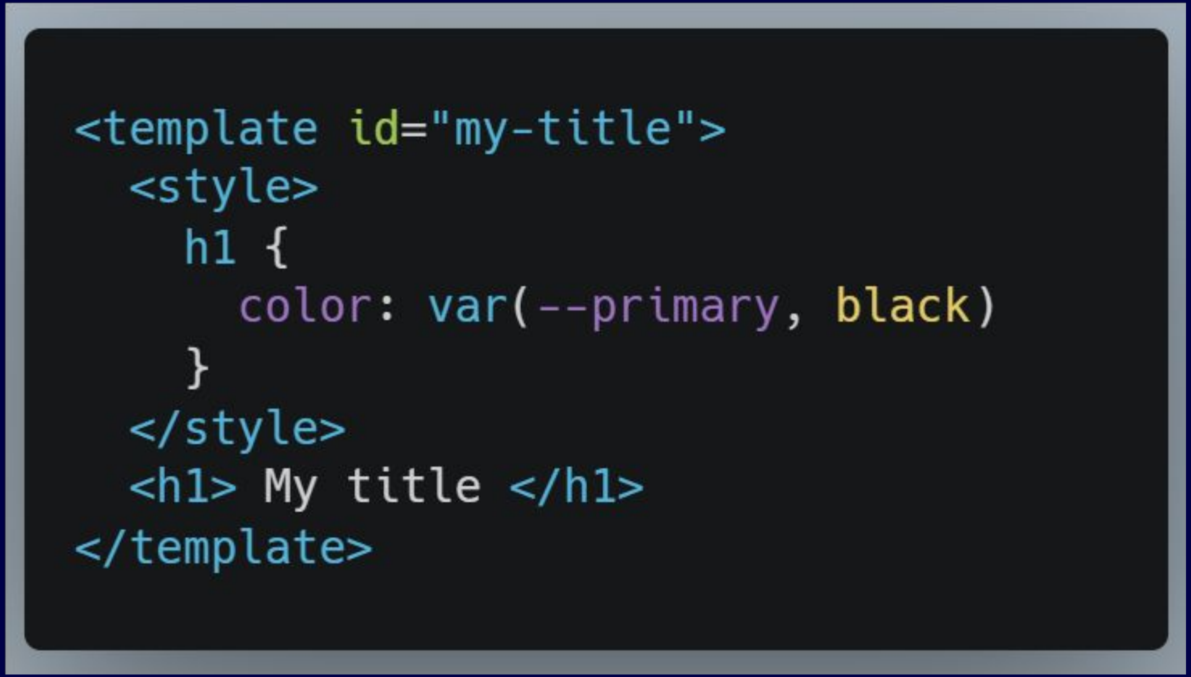
```css
prw-title {
  display: flex;
  flex-direction: column;
  width: 500px;
}

prw-title h1 {
  width: 250px;
}
```



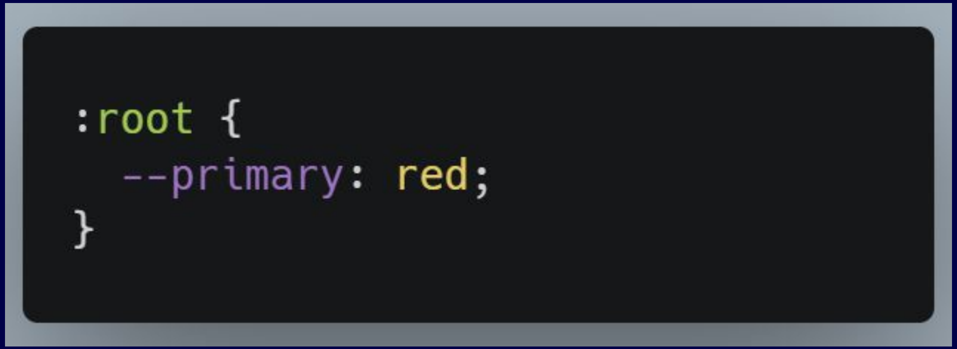
```html
<prw-title> <h1> Hello world </h1> </prw-title>
```

cognizant
softvision

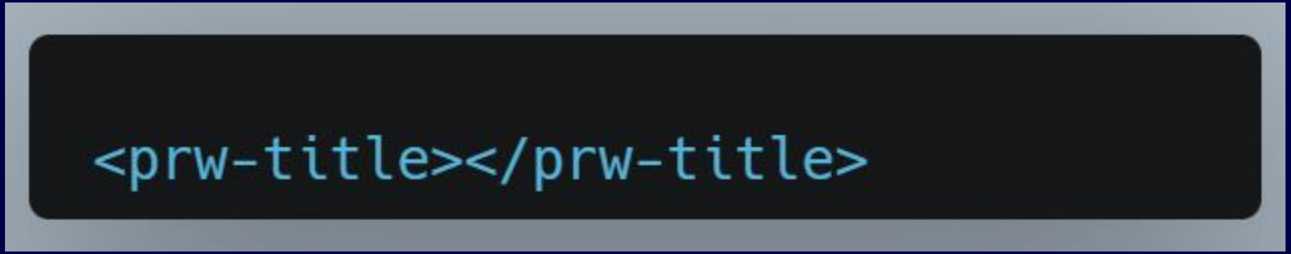
CSS variables



```
<template id="my-title">
  <style>
    h1 {
      color: var(--primary, black)
    }
  </style>
  <h1> My title </h1>
</template>
```



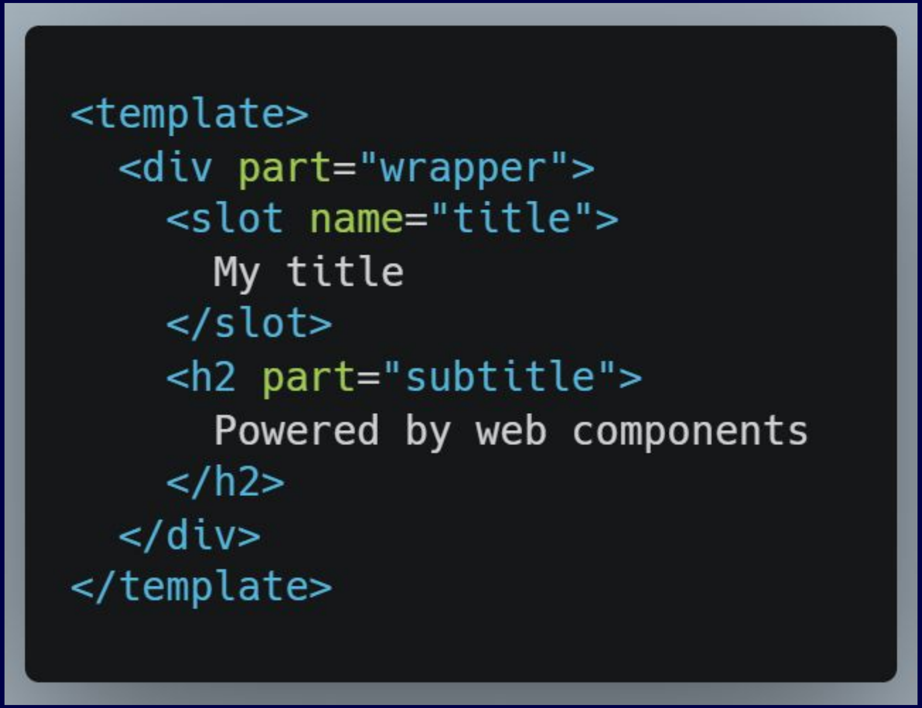
```
:root {
  --primary: red;
}
```



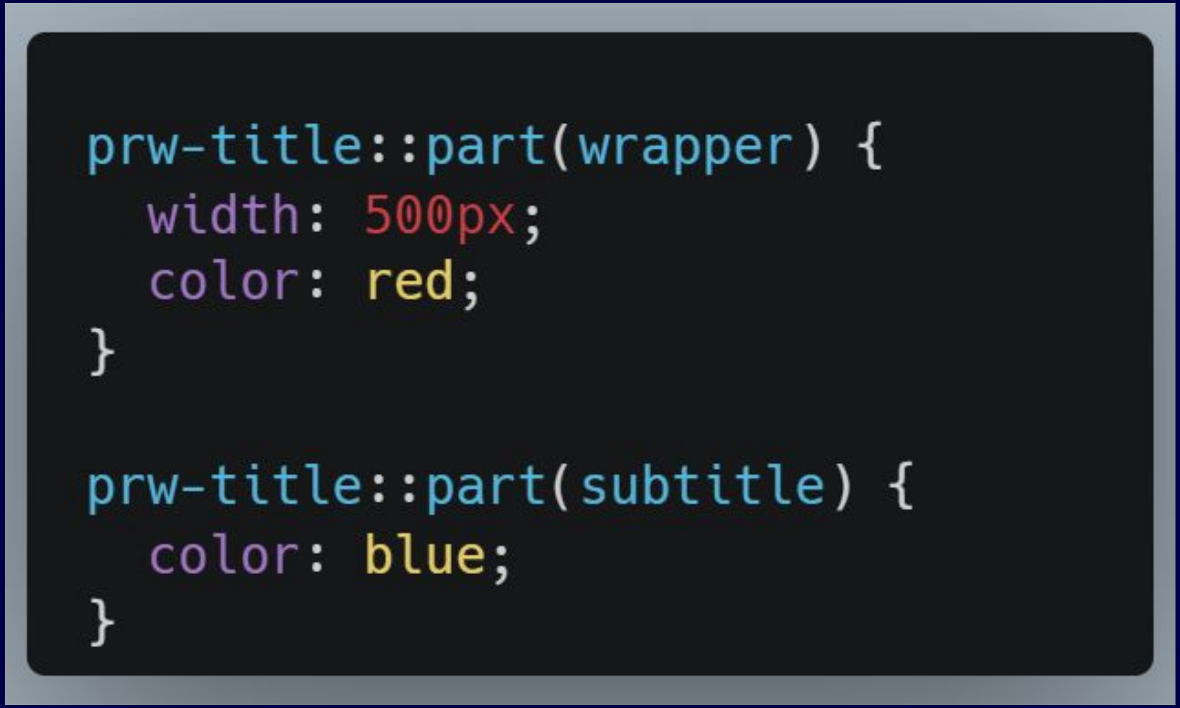
```
<prw-title></prw-title>
```

::part pseudo element

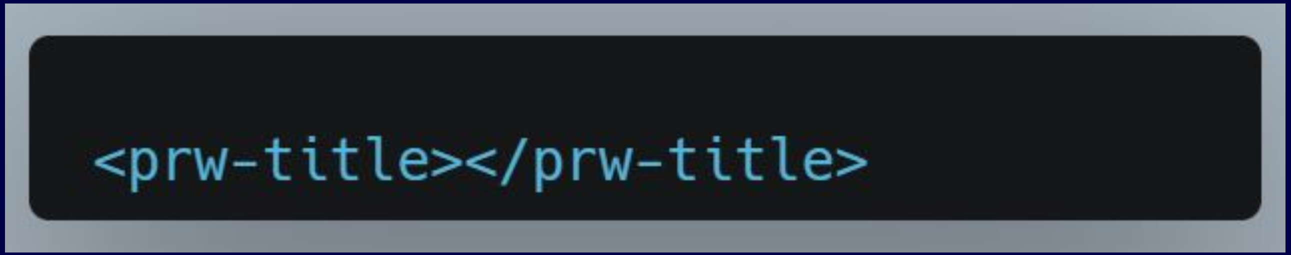
PROGRAMMERS' WEEK 2022



```html
<template>
  <div part="wrapper">
    <slot name="title">
      My title
    </slot>
    <h2 part="subtitle">
      Powered by web components
    </h2>
  </div>
</template>
```



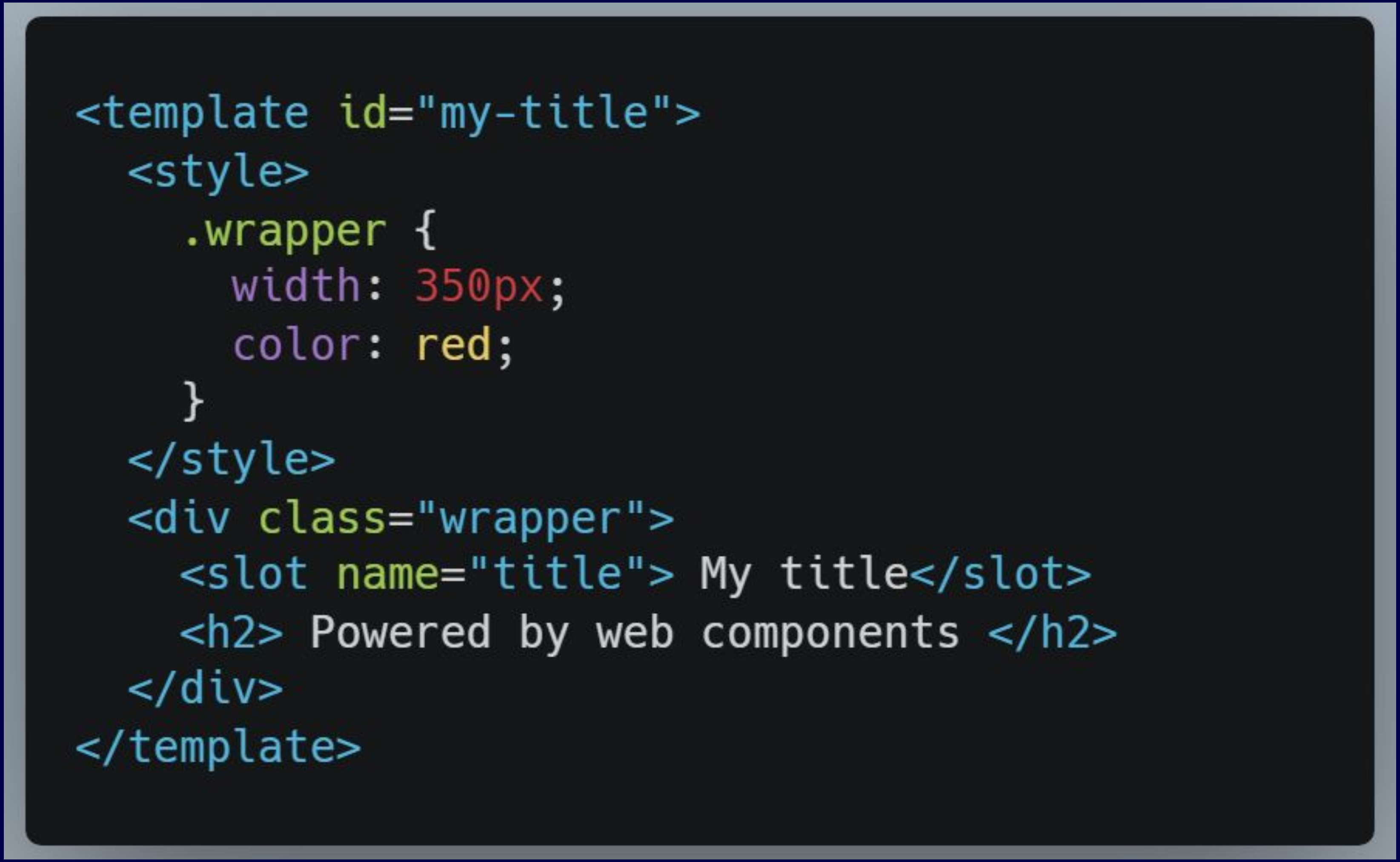
```css
prw-title::part(wrapper) {
  width: 500px;
  color: red;
}

prw-title::part(subtitle) {
  color: blue;
}
```



```html
<prw-title></prw-title>
```

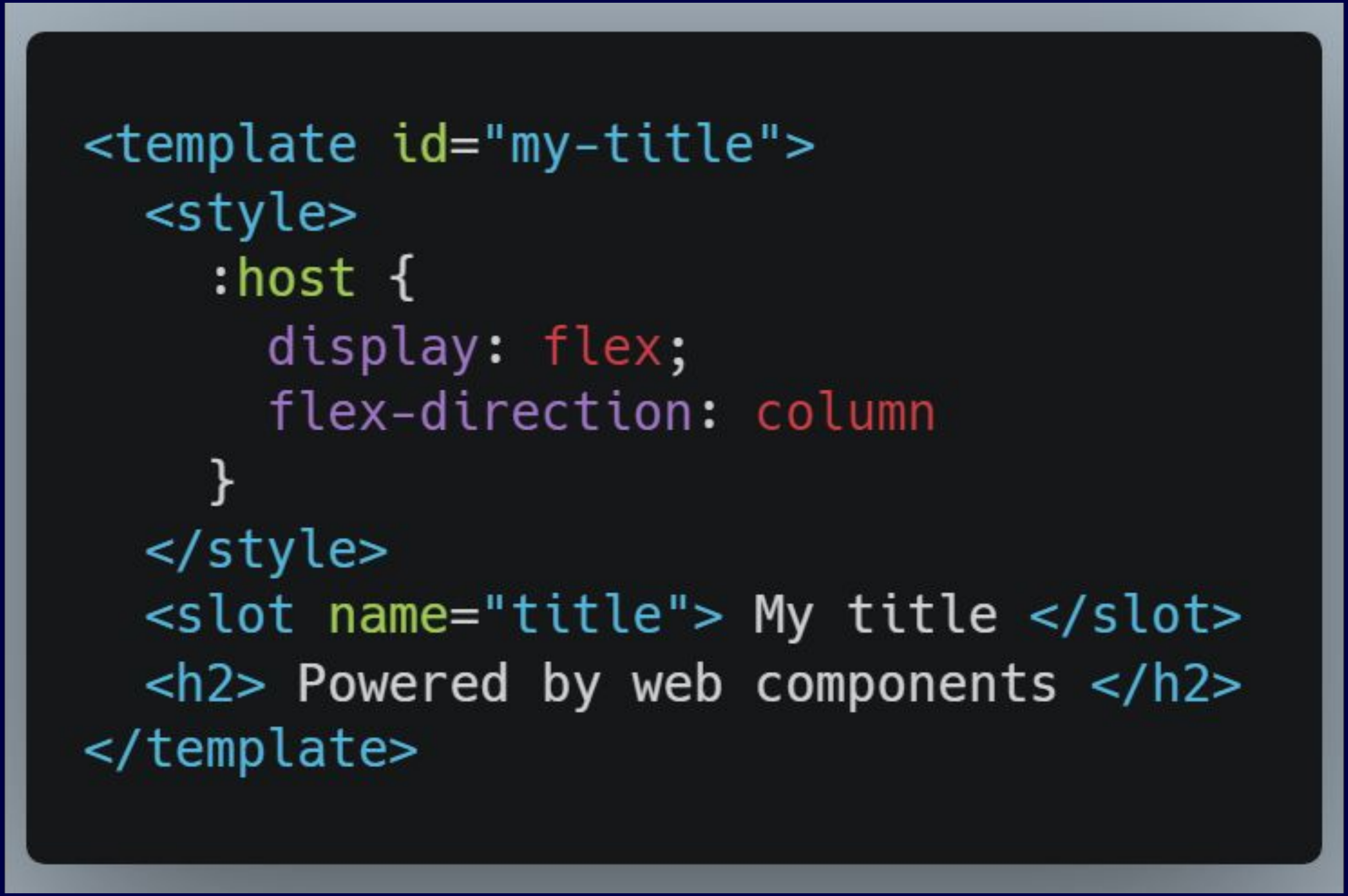
cognizant softvision

Inline style



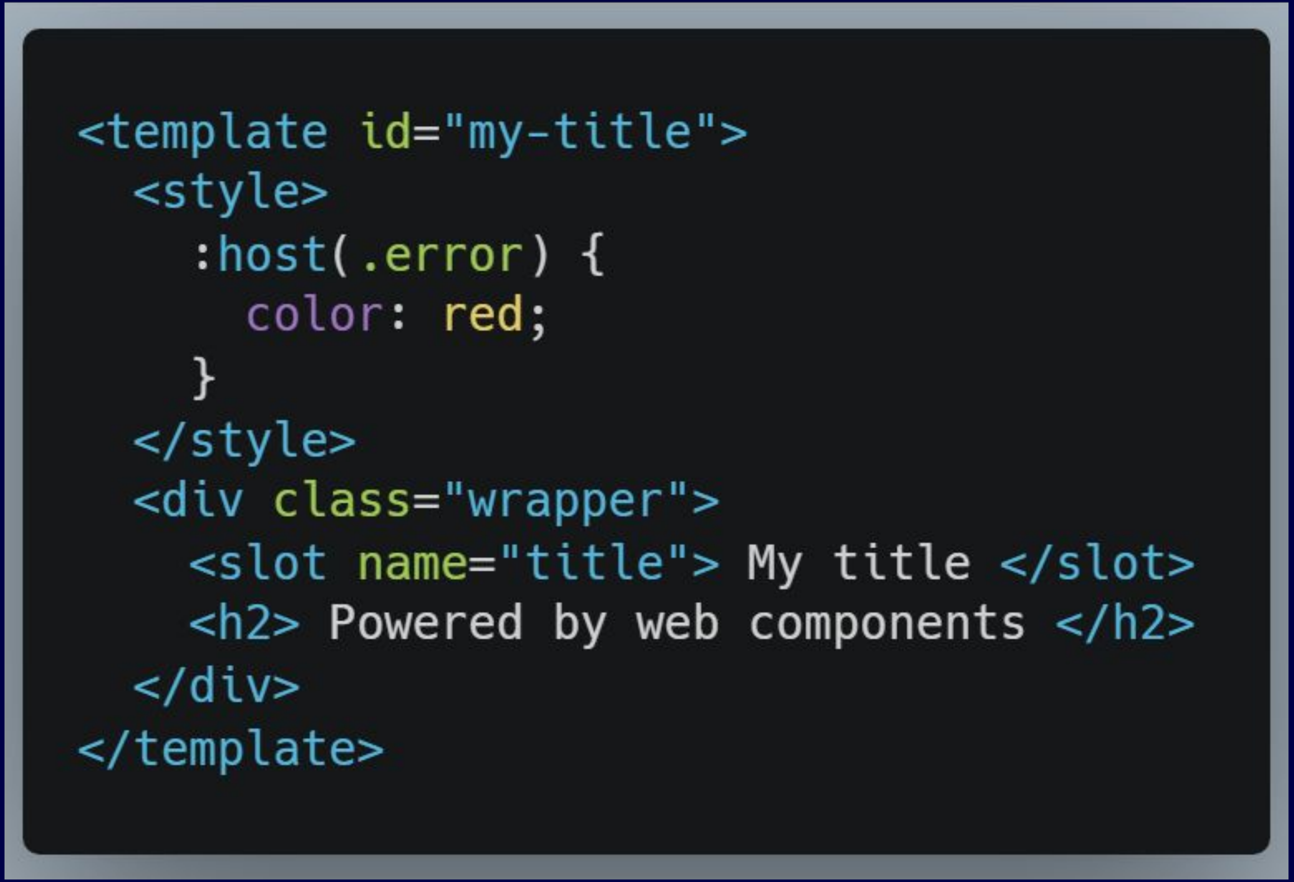
```
<template id="my-title">
  <style>
    .wrapper {
      width: 350px;
      color: red;
    }
  </style>
  <div class="wrapper">
    <slot name="title"> My title</slot>
    <h2> Powered by web components </h2>
  </div>
</template>
```

:host pseudo class

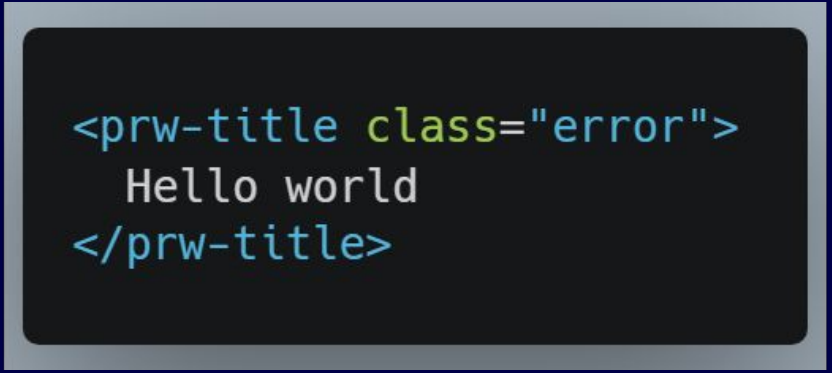


```
<template id="my-title">
  <style>
    :host {
      display: flex;
      flex-direction: column
    }
  </style>
  <slot name="title"> My title </slot>
  <h2> Powered by web components </h2>
</template>
```

::host() function

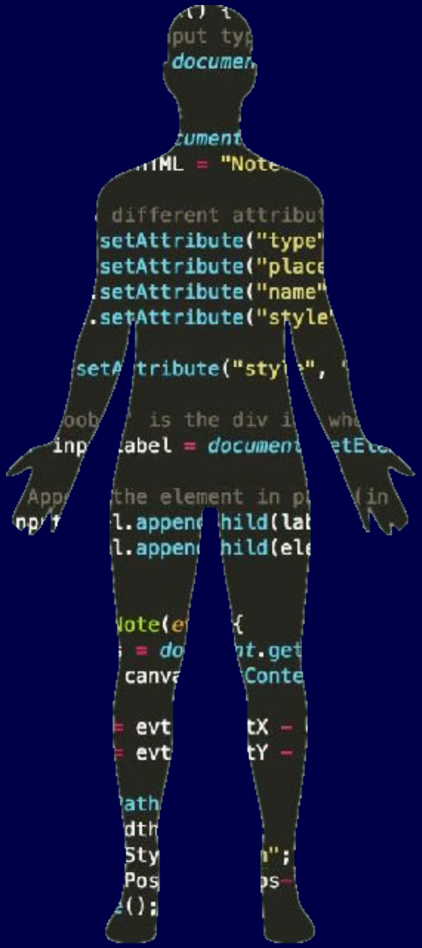


```html
<template id="my-title">
  <style>
    :host(.error) {
      color: red;
    }
  </style>
  <div class="wrapper">
    <slot name="title"> My title </slot>
    <h2> Powered by web components </h2>
  </div>
</template>
```

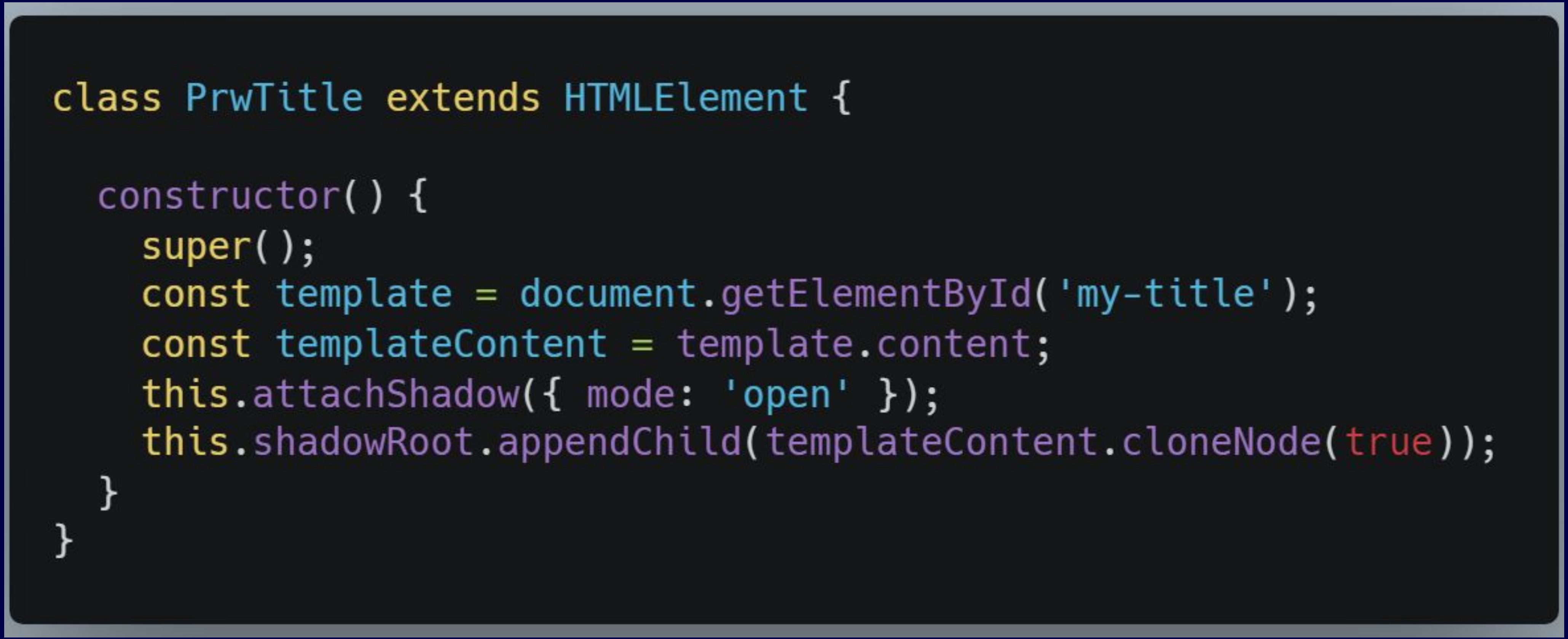


```html
<prw-title class="error">
  Hello world
</prw-title>
```

Anatomy of a web component



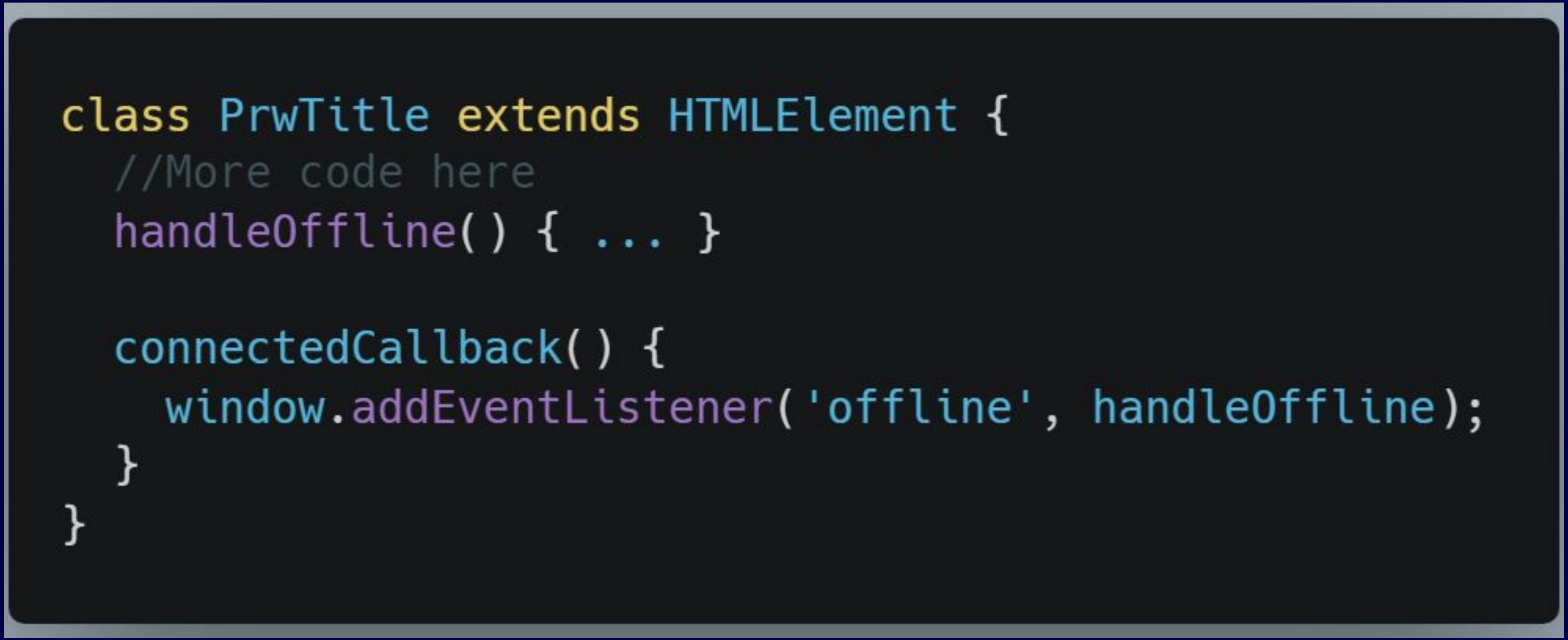
Constructor



```
class PrwTitle extends HTMLElement {

  constructor() {
    super();
    const template = document.getElementById('my-title');
    const templateContent = template.content;
    this.attachShadow({ mode: 'open' });
    this.shadowRoot.appendChild(templateContent.cloneNode(true));
  }
}
```

ConnectedCallback method

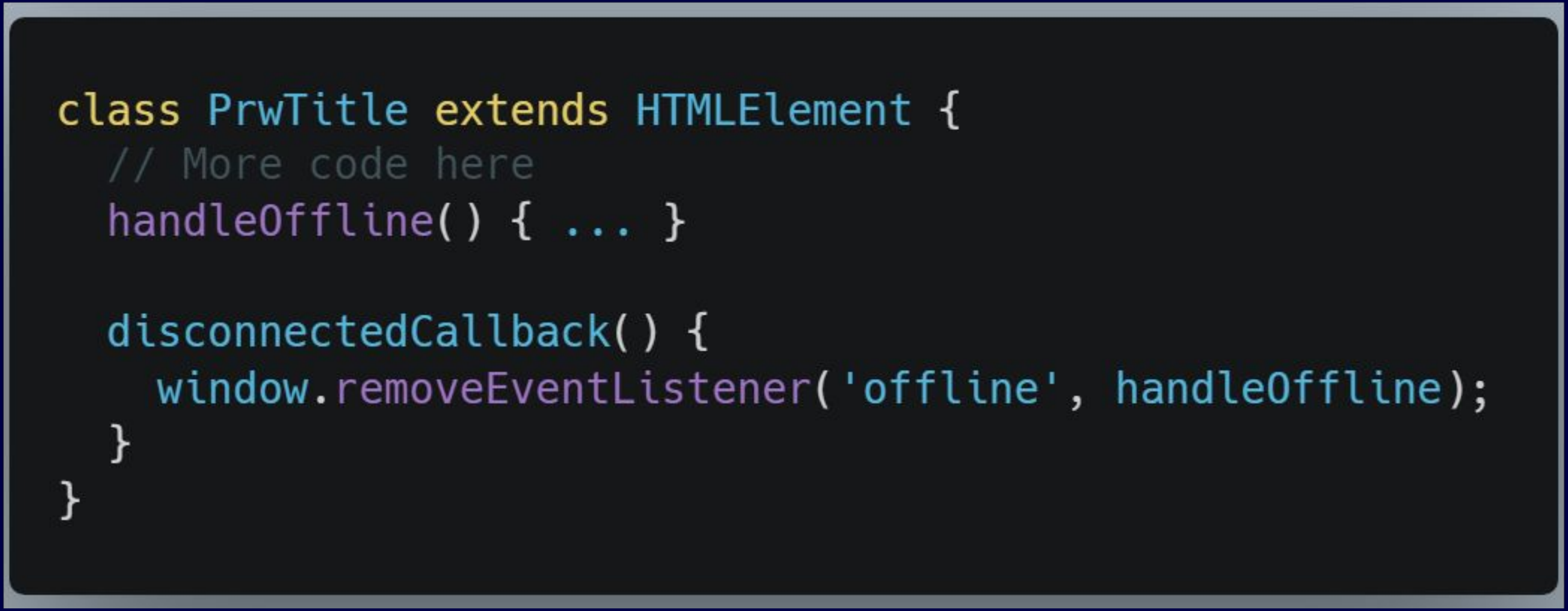
PROGRAMMERS´
WEEK 2022



```
class PrwTitle extends HTMLElement {
  //More code here
  handleOffline() { ... }

  connectedCallback() {
    window.addEventListener('offline', handleOffline);
  }
}
```

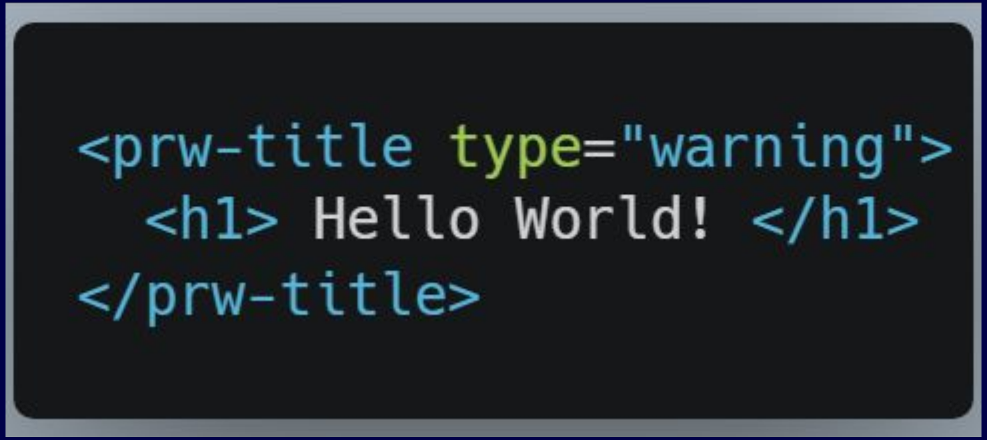
cognizant
softvision

DisconnectedCallback method

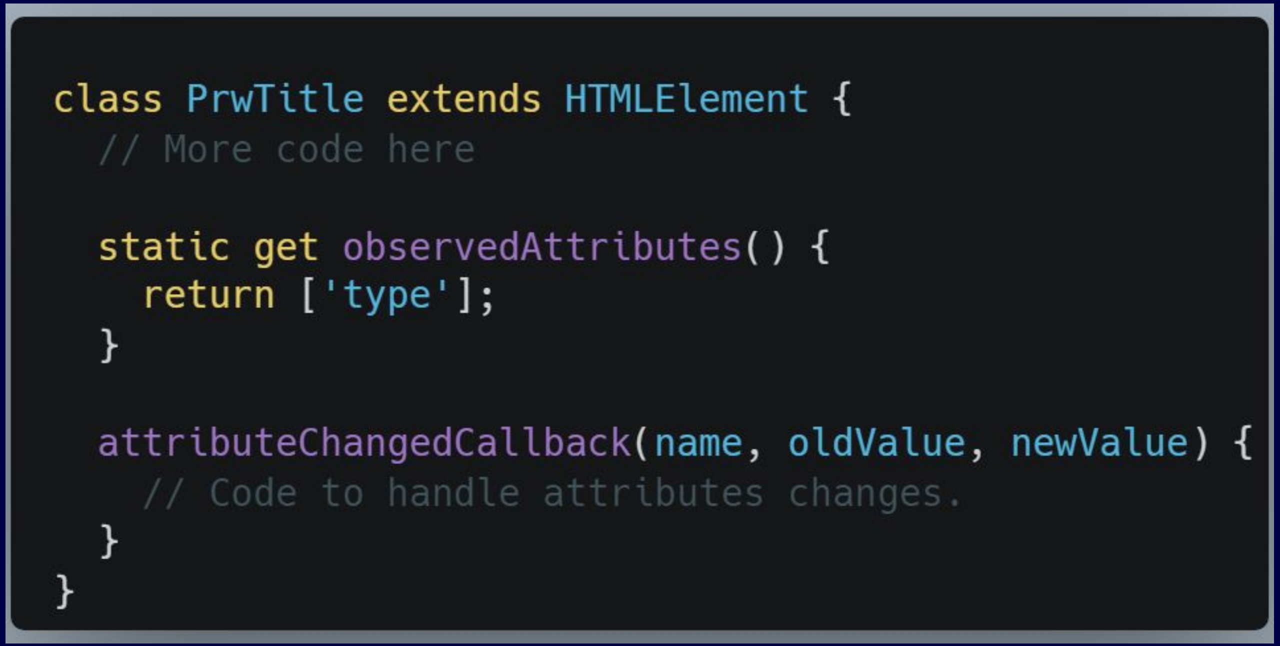


```
class PrwTitle extends HTMLElement {
  // More code here
  handleOffline() { ... }

  disconnectedCallback() {
    window.removeEventListener('offline', handleOffline);
  }
}
```

AttributeChangedCallback method

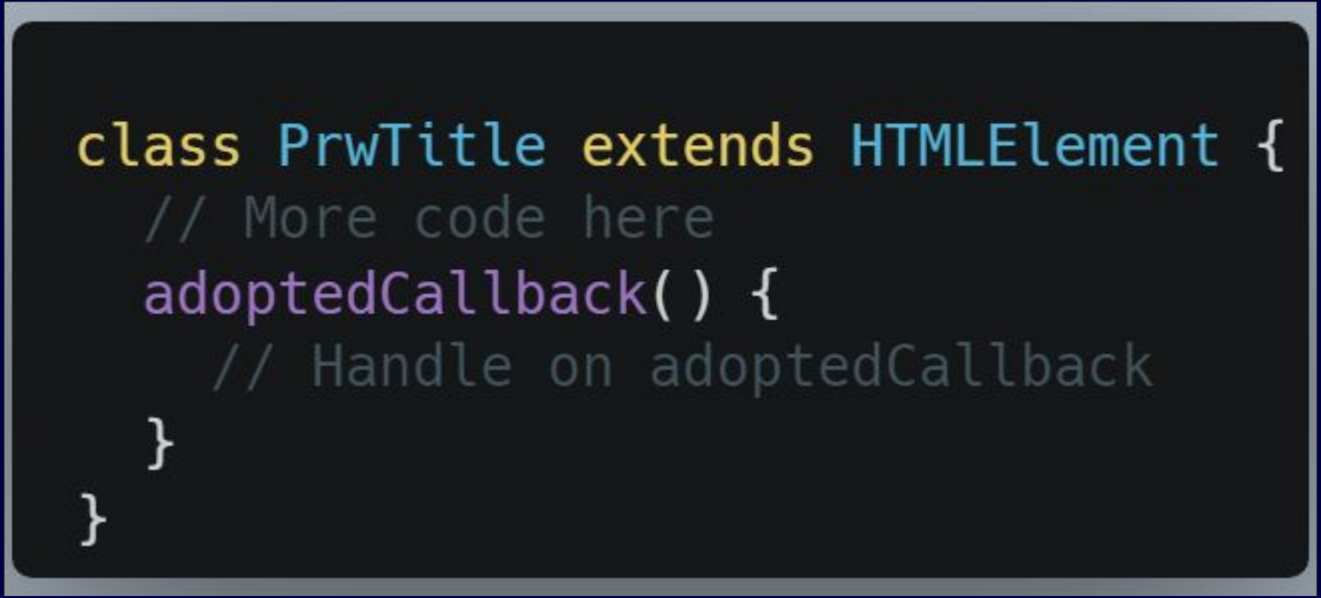


```html
<prw-title type="warning">
  <h1> Hello World! </h1>
</prw-title>
```



```javascript
class PrwTitle extends HTMLElement {
  // More code here

  static get observedAttributes() {
    return ['type'];
  }

  attributeChangedCallback(name, oldValue, newValue) {
    // Code to handle attributes changes.
  }
}
```

AdoptedCallback method

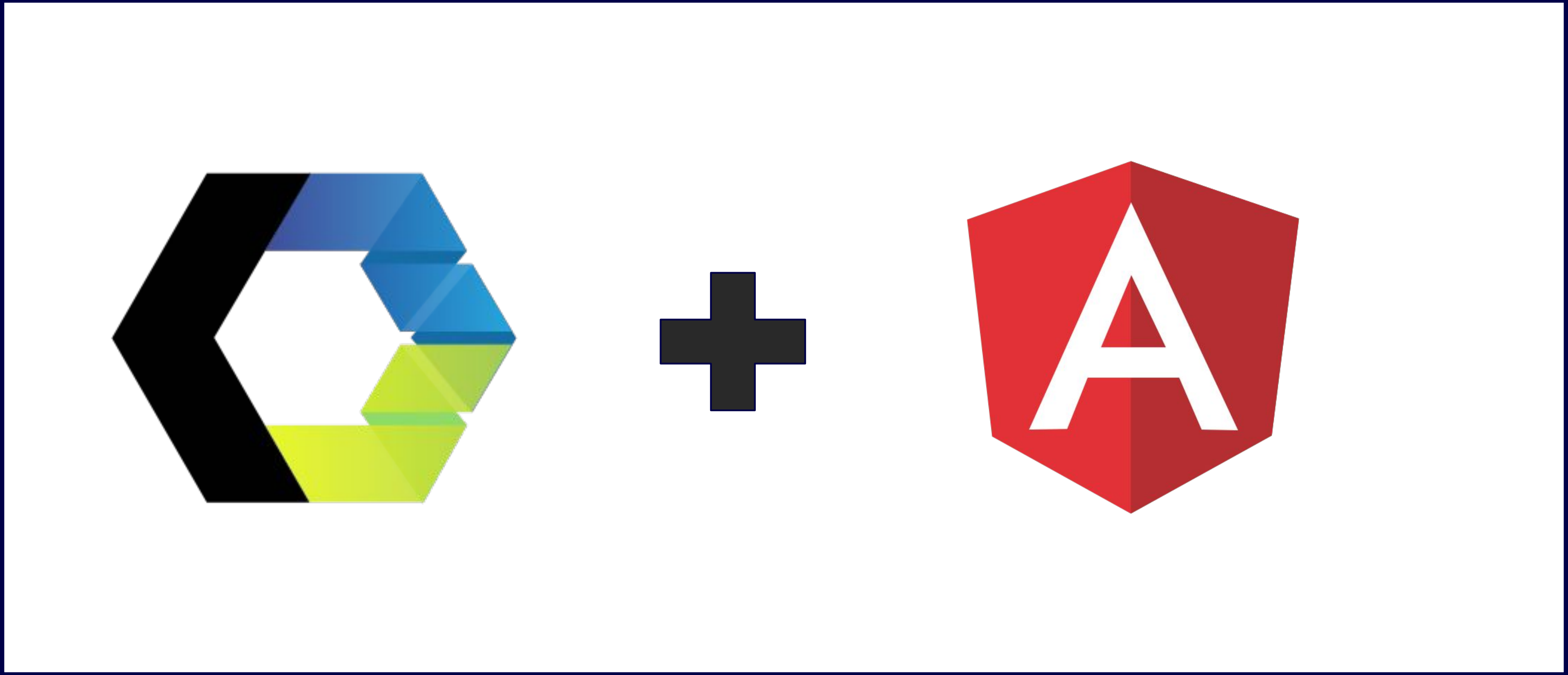


```
class PrwTitle extends HTMLElement {
  // More code here
  adoptedCallback() {
    // Handle on adoptedCallback
  }
}
```

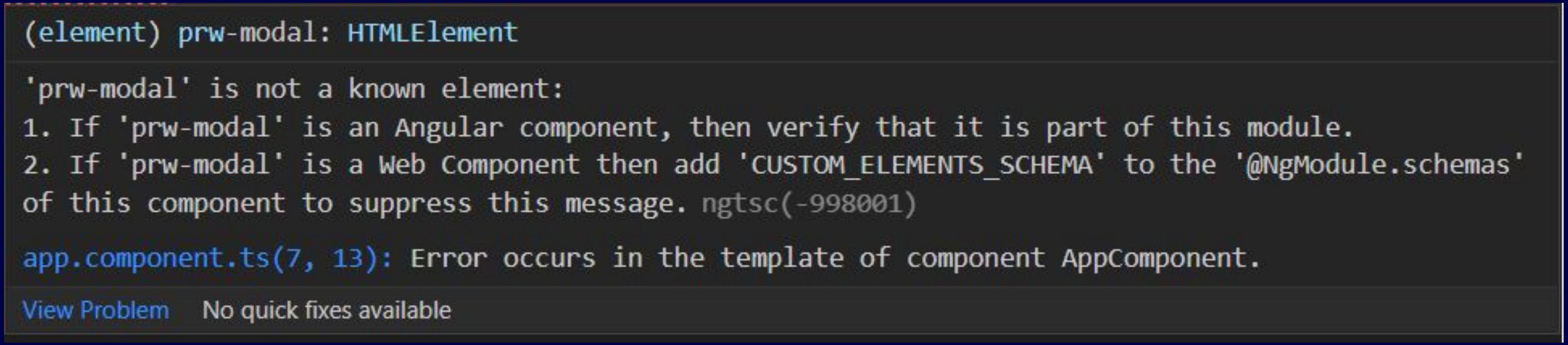
cognizant softvision | PROGRAMMERS' WEEK 2022

Working sample of a web component

Web components in Angular

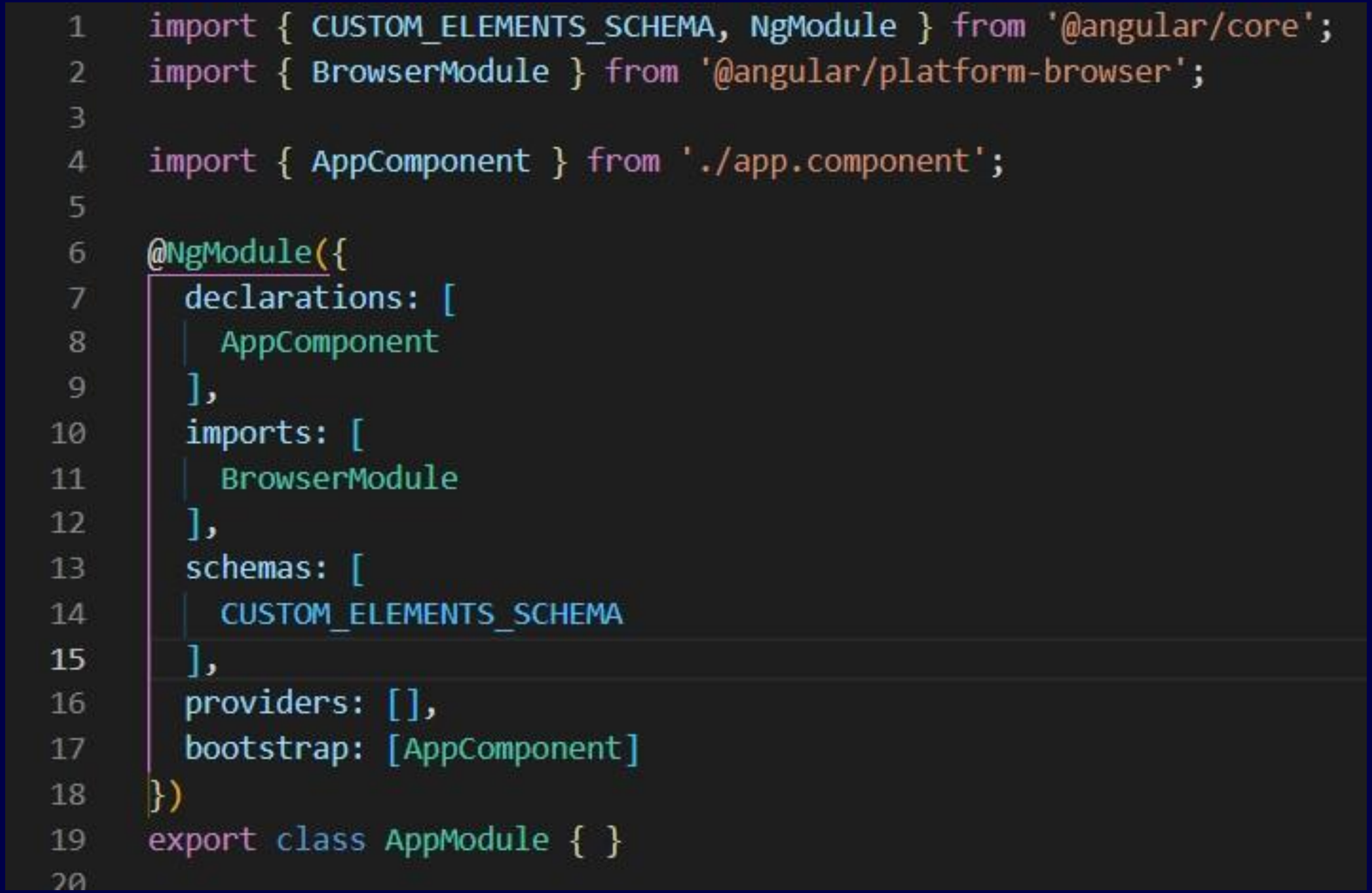


Web components in Angular



```
(element) prw-modal: HTMLElement

'prw-modal' is not a known element:
1. If 'prw-modal' is an Angular component, then verify that it is part of this module.
2. If 'prw-modal' is a Web Component then add 'CUSTOM_ELEMENTS_SCHEMA' to the '@NgModule.schemas'
of this component to suppress this message. ngtsc(-998001)

app.component.ts(7, 13): Error occurs in the template of component AppComponent.
```

Web components in Angular

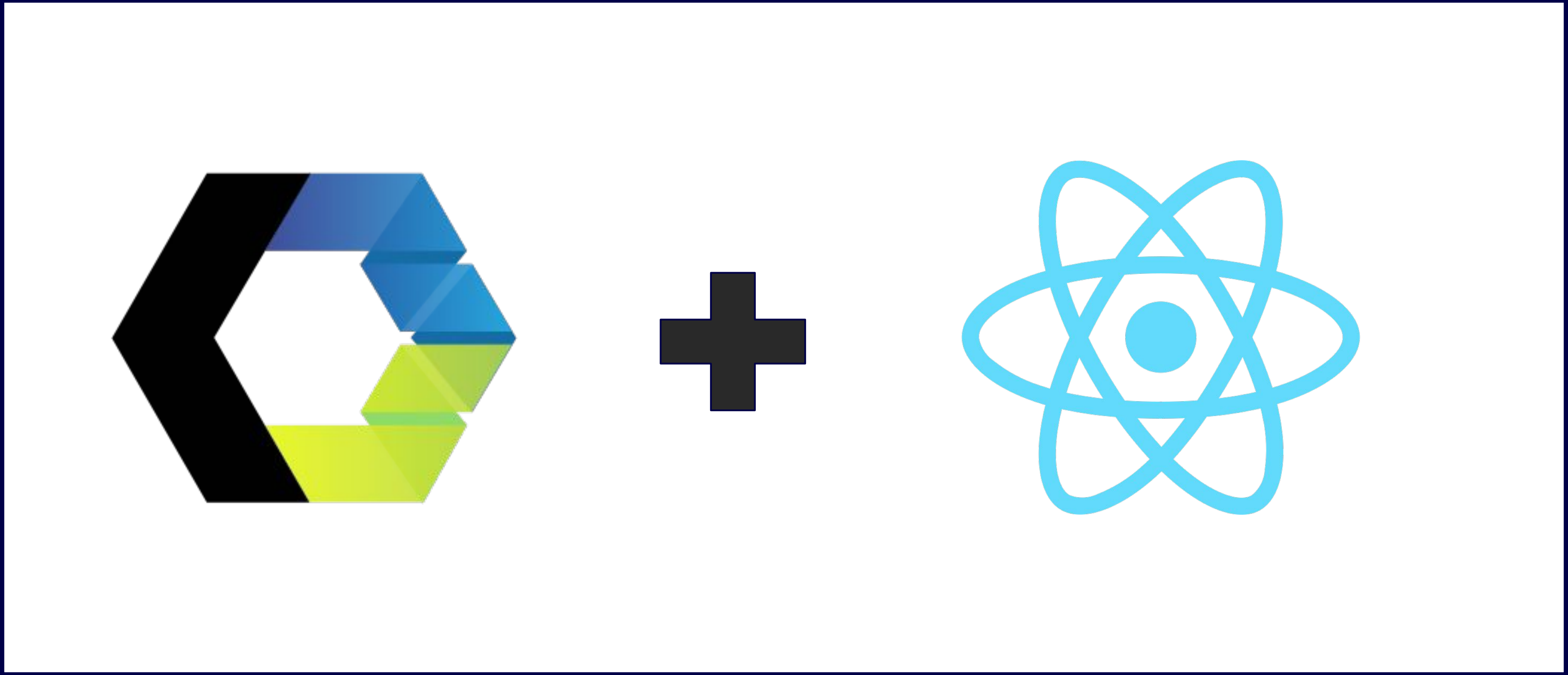


```
import { CUSTOM_ELEMENTS_SCHEMA, NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule
  ],
  schemas: [
    CUSTOM_ELEMENTS_SCHEMA
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

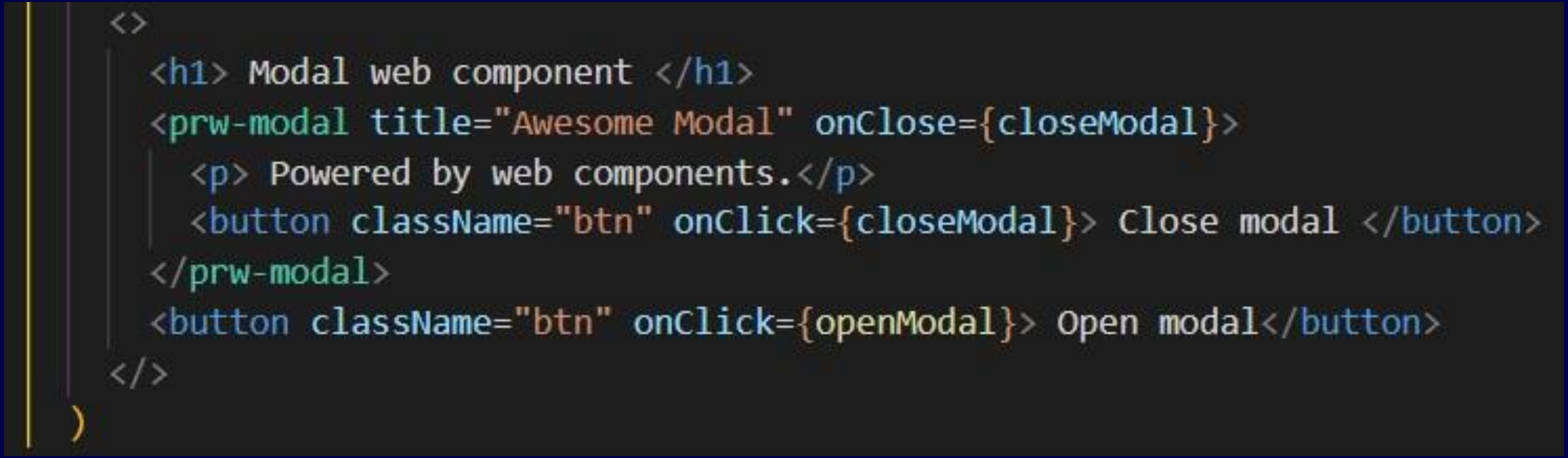
cognizant softvision | PROGRAMMERS' WEEK 2022

Working sample of a web component in Angular

Web components in React

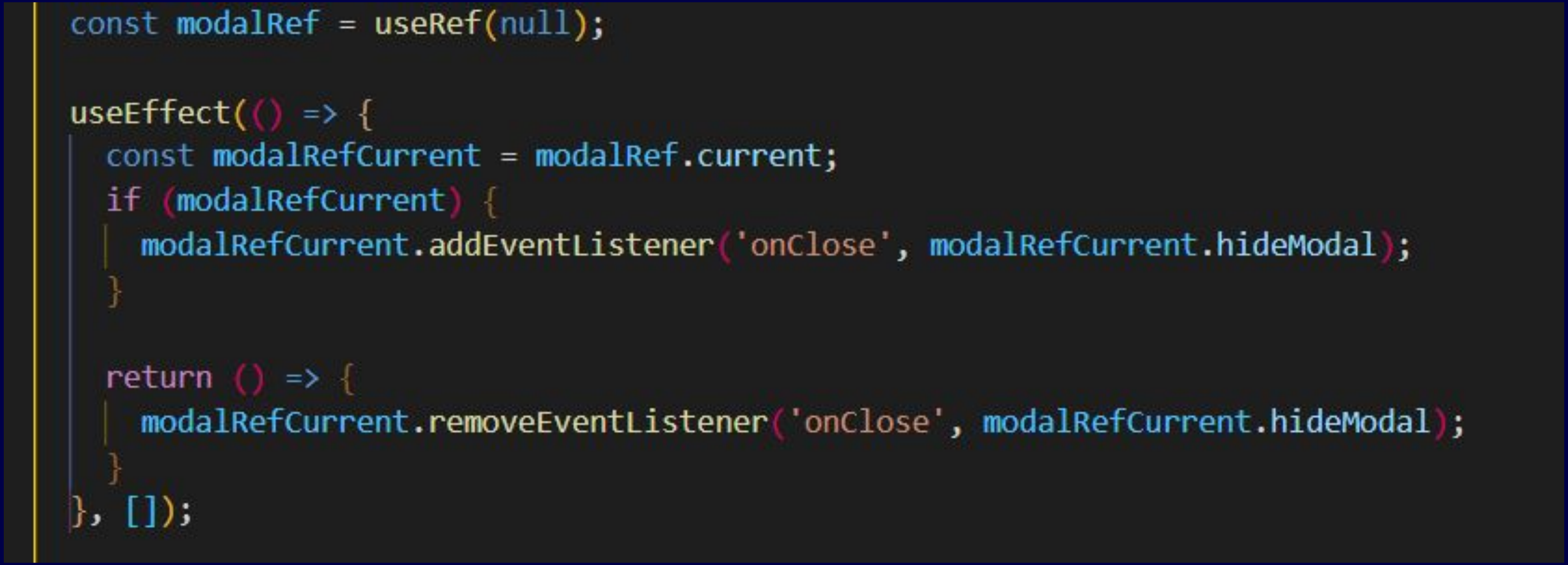


Web components in React



```
    <>
      <h1> Modal web component </h1>
      <prw-modal title="Awesome Modal" onClose={closeModal}>
        <p> Powered by web components.</p>
        <button className="btn" onClick={closeModal}> Close modal </button>
      </prw-modal>
      <button className="btn" onClick={openModal}> Open modal</button>
    </>
  )
```

Web components in React



```
const modalRef = useRef(null);

useEffect(() => {
  const modalRefCurrent = modalRef.current;
  if (modalRefCurrent) {
    modalRefCurrent.addEventListener('onClose', modalRefCurrent.hideModal);
  }

  return () => {
    modalRefCurrent.removeEventListener('onClose', modalRefCurrent.hideModal);
  }
}, []);
```

cognizant softvision | PROGRAMMERS' WEEK 2022

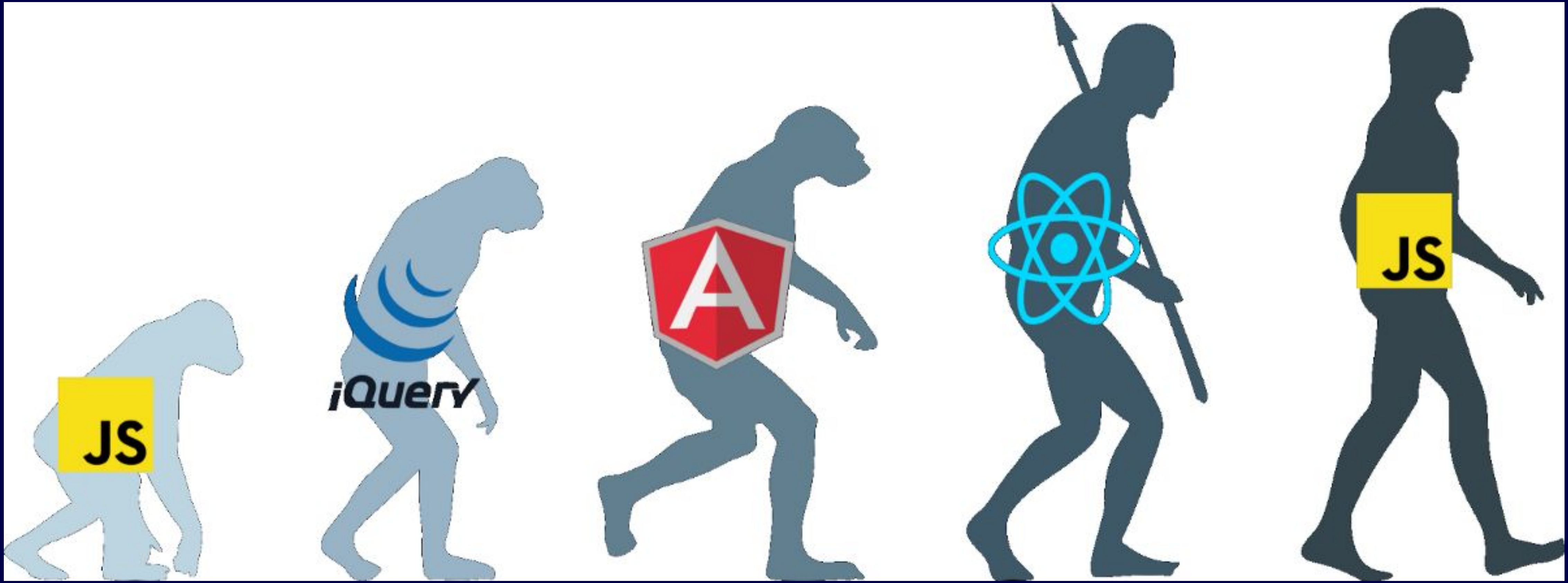
Working sample of a web component in React

cognizant softvision

PROGRAMMERS´ WEEK 2022

Final thoughts

Final thoughts



“We have to find solutions for a problem and no problems for a solution.”

Attributor name is 16pt Arial

References

PROGRAMMERS' WEEK 2022

Documentation

- Custom elements, v1, mdn
- Template
- Shadow DOM, w3c
- ShadowRoot
- Webcomponents.org, mdn

Guides, articles and more

- The state of web components in 2022
- Building components
- Custom Element Best Practices
- Book: Web Components in Action
- webcomponents.dev
- Awesome list, another list, and another

cognizant softvision

Web components repository

https://github.com/cognizant-softvision/pw2022-web-components