

Web Components

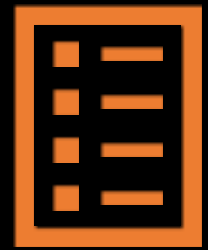
The Lego bricks of web developers

-
- **Aaron Czichon**
 - **Head of Software Development**
 - **NodeJS & Web Components**



The basic
problem of HTML
elements





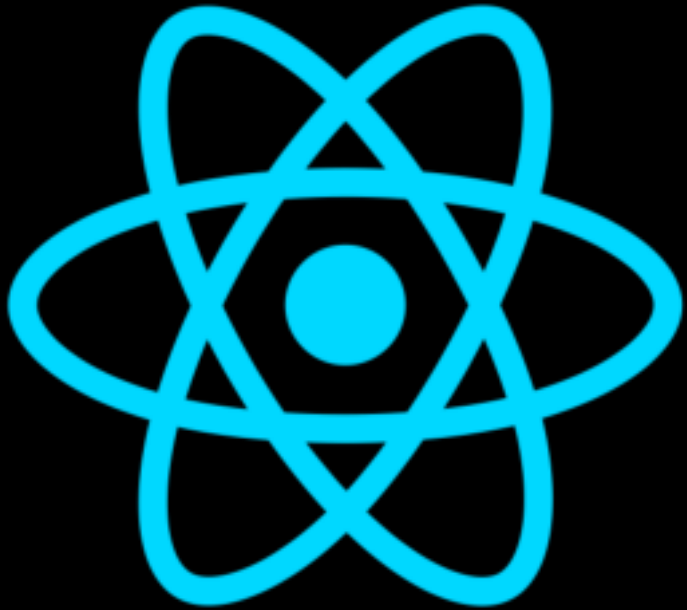
Building a login
form...

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Dev Days Europe 2019</title>
    <meta name="description" content="Example for Web Components">
    <meta name="author" content="Aaron Czichon">
    <link rel="stylesheet" href="./assets/edel-elements.css">
  </head>

  <body>
    <!-- CONTENT HERE -->
  </body>
</html>
```

```
<div class="login-card card">
  <div class="card-meta">
    <h3>Login</h3>
    <p>Please login first</p>
  </div>
  <div class="login-card-form">
    <div class="form form--grid">
      <div class="form-item">
        <label>Username or email</label>
        <input placeholder="Username or email" type="text">
      </div>
      <div class="form-item">
        <label>Password</label>
        <input placeholder="Super Secret Password" type="password">
      </div>
      <div class="form-item">
        <button type="submit" class="button button--primary">Login</button>
      </div>
    </div>
  </div>
</div>
```

Want using it in multiple applications?



Multiple Framework Options

Downsides

- **Mostly only usable inside the framework**
- **(Angular has export options for web components)**
- **Sometimes a lot of overhead which is not needed**



Web Components!



„Web components are a set of web platform APIs that allow you to create new custom, reusable, encapsulated HTML tags“

Source: webcomponents.org

Custom Elements

HTML Imports

Shadow DOM

HTML Template

Web Platform APIs

A diagram illustrating the relationship between Web Platform APIs and Web Components. At the top, a grey horizontal bar contains four yellow boxes labeled 'Custom Elements', 'HTML Imports', 'Shadow DOM', and 'HTML Template'. Below this bar, the text 'Web Platform APIs' is positioned on the left. A large yellow arrow points downwards from the 'Web Platform APIs' area towards a central yellow box labeled 'Web Component'.

Web Component

Custom Elements

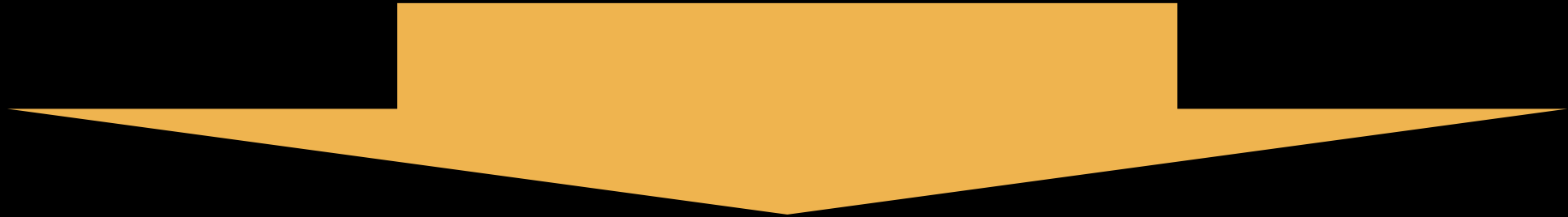
HTML Imports

Shadow DOM

HTML Template



Web Platform APIs



Web Component

Custom Elements

ES Modules

Shadow DOM

HTML Template

Web Platform APIs

A large, yellow, downward-pointing arrow with a rectangular top and a pointed bottom, centered horizontally between the 'Web Platform APIs' section and the 'Web Component' box.

Web Component

Custom Element

○○○

```
customElements.define('star-wars-character', StarWarsCharacter, { extends: 'p' });
```

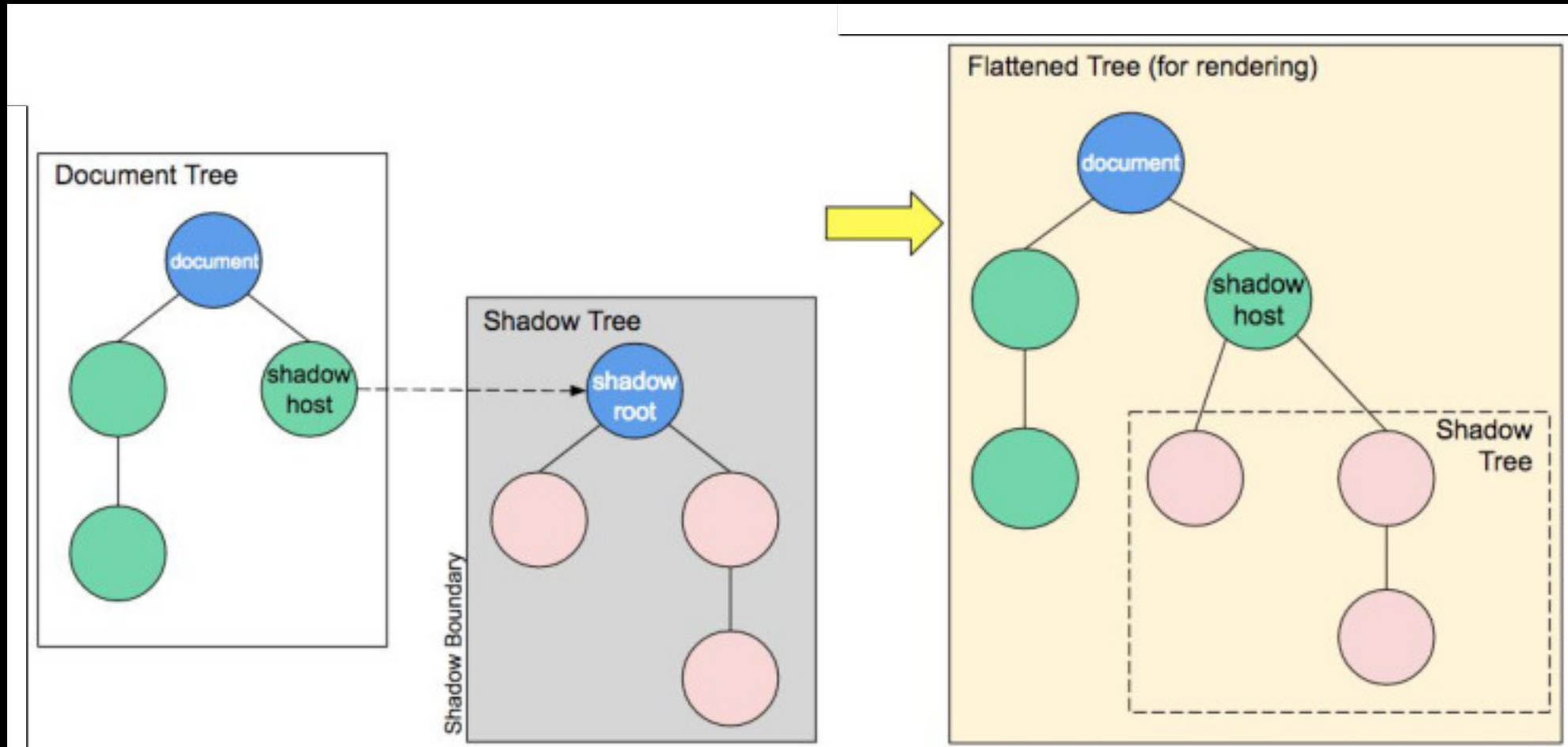
Custom Element

○○○

```
class StarWarsCharacter extends HTMLParagraphElement {  
  constructor() {  
    // Call super for trigger Paragraph constructor  
    super();  
  
    // Element implementation here  
  }  
}
```


Shadow DOM

-> Shadowed rendering of DOM Nodes



Shadow DOM

-> Attach shadow to existing element (shadow host)



```
let shadow = elementRef.attachShadow({ mode: 'open' });
```

Shadow DOM

-> Create new element and attach it to the shadowed element



```
const characterName = document.createElement('h1');  
shadow.appendChild(characterName);
```

HTML Imports

-> Importing HTMLmarkup like script files (W3C Working Draft)

○ ○ ○

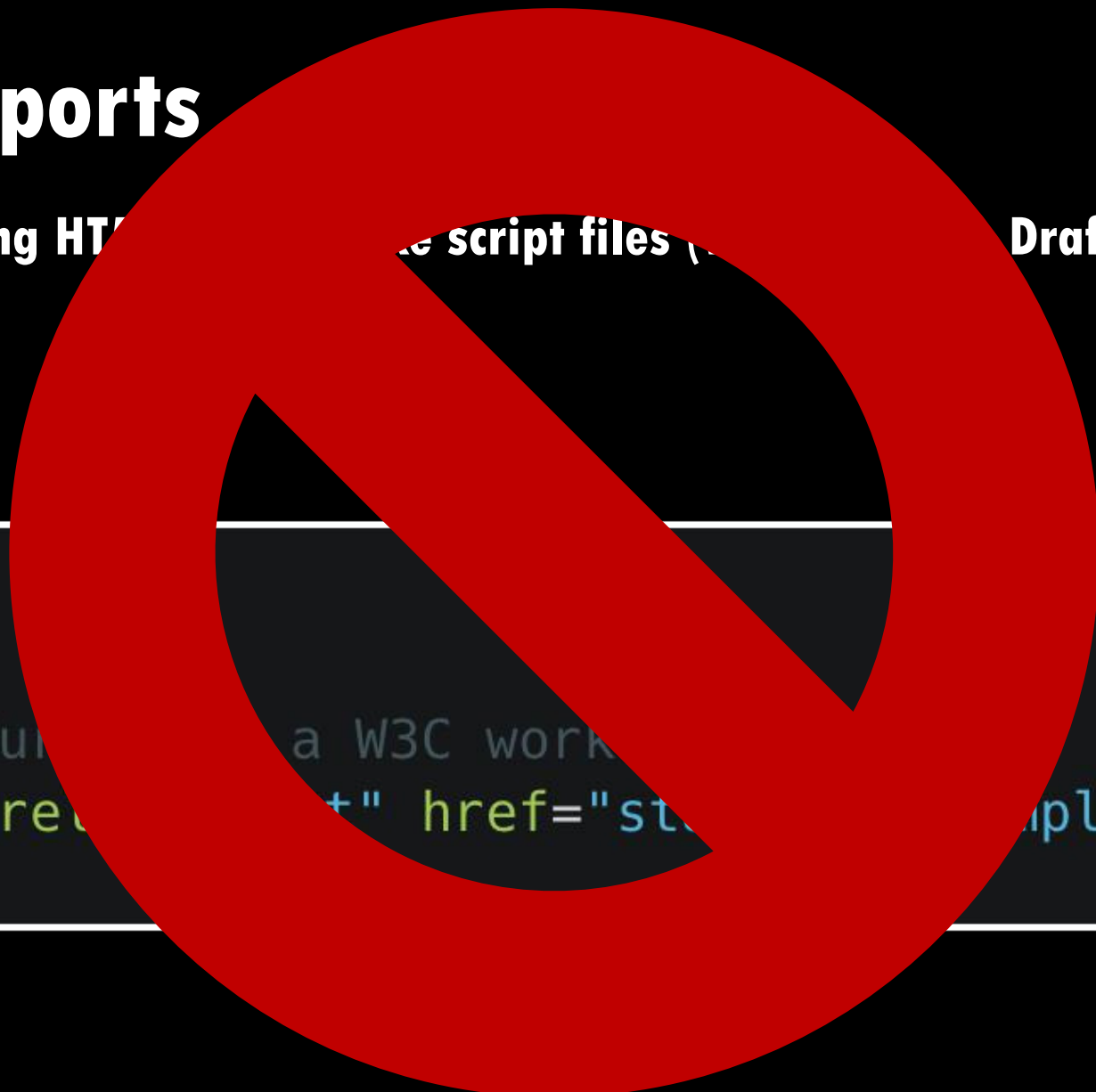
```
<!-- Currently a W3C working draft -->
```

```
<link rel="import" href="star-wars-template.html">
```

HTML Imports

-> Importing HTML files (e.g. `script files` (Draft))

```
○ ○ ○  
  
<!-- Current document is a W3C work  
<link rel="stylesheet" href="style-template.html">
```



ES Module

-> Using your module inside your website/application

○ ○ ○

```
<script type="module" src="star-wars-character.js"></script>
```

HTML Templates

-> Render templates on specific location with JS

○ ○ ○

```
<!-- Define HTML Template -->  
<template id="sw-character-component">  
  <h1>Yoda</h1>  
</template>
```

HTML Templates

-> Render templates on specific location with JS

○○○

```
// Render with JS
const template = document.getElementById('sw-character-component');
const clone = document.importNode(template.content, true);
document.body.appendChild(clone);
```


Combine everything!

-> Create Javascript component

```
○○○  
  
// component.js  
class SwCharacter extends HTMLElement {  
  constructor() {  
    super();  
    const template = document.querySelector('#star-wars-template');  
    const clone = document.importNode(template.content, true);  
    const shadowRoot = this.attachShadow({mode: 'open'});  
    shadowRoot.appendChild(clone);  
  }  
}  
  
customElements.define('sw-character-component', SwCharacter);
```

Combine everything!

-> Create Markup

```
○ ○ ○  
  
<!-- index.html -->  
<head>  
  <script src="component.js"></script>  
</head>  
  
<body>  
  <sw-character-component></sw-character-component>  
  
  <template id="star-wars-template">  
    <style> /* CSS HERE */</style>  
    <h1>Yoda</h1>  
  </template>  
</body>
```

Let's talk about browser support...

Support – HTML Templates



Support – Custom Elements



Support – Shadow DOM

Shadow DOM (V1) WD Usage % of all users Global 74.48% + 12.68% = 87.16%

Method of establishing and maintaining functional boundaries between DOM trees and how these trees interact with each other within a document, thus enabling better functional encapsulation within the DOM & CSS.

Current aligned Usage relative Date relative Apply filters Show all ?

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	BlackBerry Browser	Opera Mobile	Chrome for Android	Firefox for Android	IE Mobile	UC Browser for Android	Samsung Internet
		2-57													
		58													4
		59-62	4-52	3.1-9.1	10-39	3.2-9.3									5.4
6-10	12-17	63-65	53-73	10-12	40-57	10-12.1		2.1-4.4.4	7	12-12.1			10		6.2-
11	18	66	74	12.1	58	12.2	all	67	10	46	74	66	11	11.8	9.2
	75	67-68	75-77	TP											

Notes Known issues (0) Resources (8) Feedback

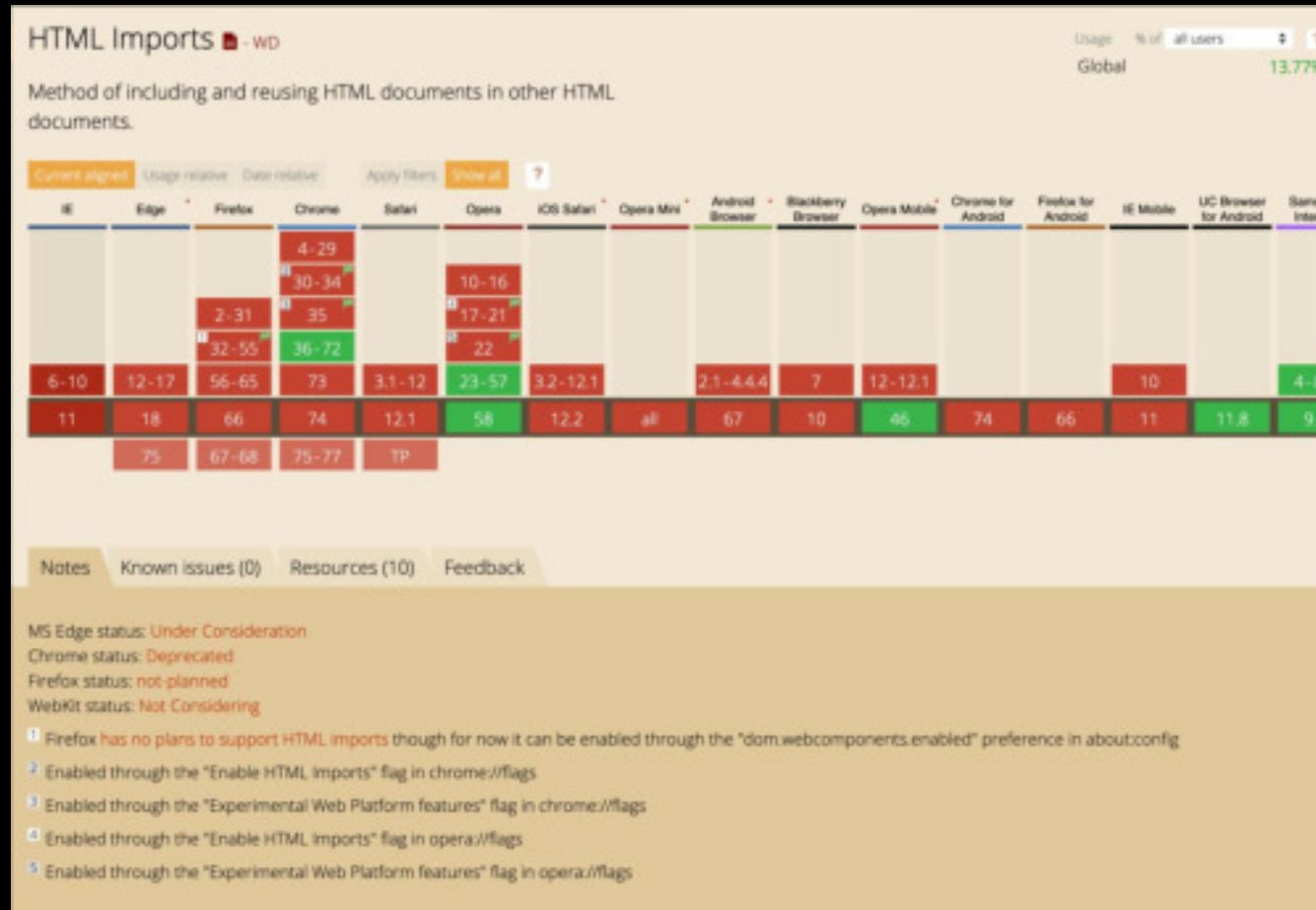
MS Edge status: *In Development*

¹ Certain CSS selectors do not work (:host > .local-child) and styling slotted content (::slotted) is buggy.

² Enabled through the dom.webcomponents.enabled preference in about:config.

³ Enabled through the dom.webcomponents.shadowdom.enabled preference in about:config.

Support – HTML Imports



Use Polyfills

- Add support with Polyfills
- This extends browsers with new features
- Downside: Needs more bandwidth (larger websites)
- -> Webcomponents.js on Github
<https://github.com/webcomponents/webcomponentsjs>

Easier development with libraries



POLYMER 2



STENCILJS



SKATEJS (WITH PREACT
OR LIT-HTML)

Web component

Polymer 2

Angular Elements

Vue-wrapper

StencilJS

```

1 class ToDoItem extends HTMLElement {
2   constructor() {
3     super();
4     this._root = this.attachShadow({ mode: 'open' });
5     this._checked = false;
6     this._index = '';
7   }
8
9   connectedCallback() {
10    this._root.innerHTML = `
11      <div>
12        <div style=
13          <div class="item">
14            <input type="checkbox" checked=${this._checked} />
15            <span style="float: right; text-align: right; font-weight: bold; font-size: 0.8em; margin-left: 10px;">
16              <button class="destroy" style="float: right; font-size: 0.7em; border: none; background: none; cursor: pointer; padding: 0 5px;">
17                ✕
18            </div>
19          </div>
20        </div>
21      `;
22
23    this._state = this._root.querySelector('.item');
24    this._removeButton = this._root.querySelector('.destroy');
25    this._state = this._root.querySelector('label');
26    this._checkbox = this._root.querySelector('input');
27    this._removeButton.addEventListener('click', () => {
28      e.preventDefault();
29      this.dispatchEvent(new CustomEvent('remove', { detail: this._index }));
30    });
31
32    this._checkbox.addEventListener('click', (e) => {
33      e.preventDefault();
34      this.dispatchEvent(new CustomEvent('toggle', { detail: this._index }));
35    });
36
37    this._render();
38
39    disconnectedCallback() {}
40
41    static get observedAttributes() {
42      return ['checked'];
43    }
44
45    attributeChangedCallback(name, oldValue, newValue) {
46      this._checked = newValue;
47    }
48
49    get index() {
50      return this._index;
51    }
52
53    get checked() {
54      return this._checked;
55    }
56
57    get checked() {
58      return this._checked;
59    }
60
61    _render() {
62      if (this._state) return;
63      this._state.textContent = this._checked ? 'Completed' : '';
64      this._checkbox.setAttribute('checked', this._checked);
65    }
66
67    _remove() {
68      this._state.textContent = 'Completed';
69      this._checkbox.setAttribute('checked', true);
70    }
71
72    _toggle() {
73      this._checkbox.setAttribute('checked', !this._checked);
74    }
75
76    _destroy() {
77      this._removeButton.remove();
78    }
79
80    _state;
81    _removeButton;
82    _checkbox;
83    _index;
84    _checked;
85  }
86
87 window.customElements.define('todo-item', ToDoItem);

```

```

1 <!-- @polymer/elements -->
2 <!-- @polymer/elements/polymer/polymer-element.html -->
3
4 <div class="item">
5   <div style=
6     <div class="item">
7       <input type="checkbox" checked=${checked} />
8       <span style="float: right; text-align: right; font-weight: bold; font-size: 0.8em; margin-left: 10px;">
9         <button class="destroy" style="float: right; font-size: 0.7em; border: none; background: none; cursor: pointer; padding: 0 5px;">
10           ✕
11       </span>
12     </div>
13   </div>
14
15   <script>
16     class ToDoItem extends Polymer.Element {
17       static get is() { return 'todo-item'; }
18       static get properties() {
19         return {
20           checked: {
21             type: Boolean,
22             value: false
23           },
24           index: {
25             type: Number,
26             value: ''
27           },
28           text: {
29             type: String,
30             value: ''
31           }
32         };
33       }
34
35       handleRemove(e) {
36         this.dispatchEvent(new CustomEvent('remove', { detail: this._index }));
37       }
38
39       handleToggle(e) {
40         this.dispatchEvent(new CustomEvent('toggle', { detail: this._index }));
41       }
42
43       handleCheck(e) {
44         this._checked = e.target.checked;
45       }
46
47       handleDestroy(e) {
48         this._removeButton.remove();
49       }
50
51       _render() {
52         if (this._state) return;
53         this._state.textContent = this._checked ? 'Completed' : '';
54         this._checkbox.setAttribute('checked', this._checked);
55       }
56
57       _remove() {
58         this._state.textContent = 'Completed';
59         this._checkbox.setAttribute('checked', true);
60       }
61
62       _toggle() {
63         this._checkbox.setAttribute('checked', !this._checked);
64       }
65
66       _destroy() {
67         this._removeButton.remove();
68       }
69
70       _state;
71       _removeButton;
72       _checkbox;
73       _index;
74       _checked;
75     }
76
77 window.customElements.define(ToDoItem.is, ToDoItem);

```

```

1 import { Component, EventEmitter, Input, Output, ViewEncapsulation } from '@angular/core';
2
3 @Component({
4   selector: 'todo-item',
5   template: `
6     <div class="item">
7       <input type="checkbox" checked=${checked} />
8       <span style="float: right; text-align: right; font-weight: bold; font-size: 0.8em; margin-left: 10px;">
9         <button class="destroy" style="float: right; font-size: 0.7em; border: none; background: none; cursor: pointer; padding: 0 5px;">
10           ✕
11       </span>
12     </div>
13   `,
14   styleUrls: [],
15   encapsulation: ViewEncapsulation.None
16 })
17 export class ToDoItem {
18   @Input() checked: boolean;
19   @Input() text: string;
20   @Input() index: number;
21   @Output() onToggleChecked = new EventEmitter<number>();
22   @Output() onToggleRemove = new EventEmitter<number>();
23
24   handleRemove = () => this.onToggleRemove.emit(this.index);
25   handleCheck = () => this.onToggleChecked.emit(this.index);
26
27 // @ts-ignore
28 import { props } from '@skatejs/props';
29 import { h } from 'preact';
30 import { Component } from './util';
31
32 export default class extends Component {
33   static events = ['check', 'remove'];
34   static props = {
35     checked: props.boolean,
36     index: props.number
37   };
38
39   handleCheck = e => {
40     this.onCheck({ index: this.index, value: e.target.checked });
41   };
42
43   handleRemove = () => {
44     this.onRemove({ index: this.index });
45   };
46
47   render() {
48     return (
49       <div>
50         <div style=
51           <div class="item">
52             <input type="checkbox" checked=${checked} />
53             <span style="float: right; text-align: right; font-weight: bold; font-size: 0.8em; margin-left: 10px;">
54               <button class="destroy" style="float: right; font-size: 0.7em; border: none; background: none; cursor: pointer; padding: 0 5px;">
55                 ✕
56             </span>
57           </div>
58         </div>
59       </div>
60     );
61   }
62 }

```

SkateJS + Preact

SkateJS + lit-html

```

1 <template>
2   <div class="item">
3     <input type="checkbox" checked=${checked} />
4     <span style="float: right; text-align: right; font-weight: bold; font-size: 0.8em; margin-left: 10px;">
5       <button class="destroy" style="float: right; font-size: 0.7em; border: none; background: none; cursor: pointer; padding: 0 5px;">
6         ✕
7     </span>
8   </div>
9 </template>
10
11 <script>
12   module.exports = {
13     name: 'todo-item',
14     props: ['checked', 'text', 'checked'],
15     methods: {
16       handleRemove() {
17         this.emit('remove', this.index);
18       },
19       handleToggle() {
20         this.emit('toggle', this.index);
21       }
22     }
23   };
24 </script>

```

```

1 import { Component, Prop, Event, EventEmitter } from '@stencil/core';
2
3 @Component({
4   tag: 'todo-item',
5   styleUrl: 'todo-item.scss',
6   shadow: true,
7 })
8 export class ToDoItem {
9   @Prop() checked: boolean;
10  @Prop() text: string;
11  @Prop() index: number;
12  @Event() onToggleChecked: EventEmitter;
13  @Event() onToggleRemove: EventEmitter;
14
15  handleRemove = () => this.onToggleRemove.emit(this.index);
16  handleCheck = () => this.onToggleChecked.emit(this.index);
17
18  render() {
19    return (
20      <div>
21        <div class="item">
22          <input type="checkbox" checked=${checked} />
23          <span style="float: right; text-align: right; font-weight: bold; font-size: 0.8em; margin-left: 10px;">
24            <button class="destroy" style="float: right; font-size: 0.7em; border: none; background: none; cursor: pointer; padding: 0 5px;">
25              ✕
26          </span>
27        </div>
28      </div>
29    );
30  }
31 }

```

```

1 import { props } from '@skatejs/props';
2 import { h, lit } from 'lit-html/lit-extended';
3 import { Component } from './util';
4
5 export default class extends Component {
6   static events = ['check', 'remove'];
7   static props = {
8     checked: props.boolean,
9     index: props.number
10  };
11
12  handleCheck = e => {
13    this.onCheck({ index: this.index, value: e.target.checked });
14  };
15
16  handleRemove = () => {
17    this.onRemove({ index: this.index });
18  };
19
20  render() {
21    return (
22      <div>
23        <div style=
24          <div class="item">
25            <input type="checkbox" checked=${checked} />
26            <span style="float: right; text-align: right; font-weight: bold; font-size: 0.8em; margin-left: 10px;">
27              <button class="destroy" style="float: right; font-size: 0.7em; border: none; background: none; cursor: pointer; padding: 0 5px;">
28                ✕
29            </span>
30          </div>
31        </div>
32      </div>
33    );
34  }
35 }

```

Building the login with Web Components

How we use it for



How we use it

**@edel-elements (UI-
Component Library)**

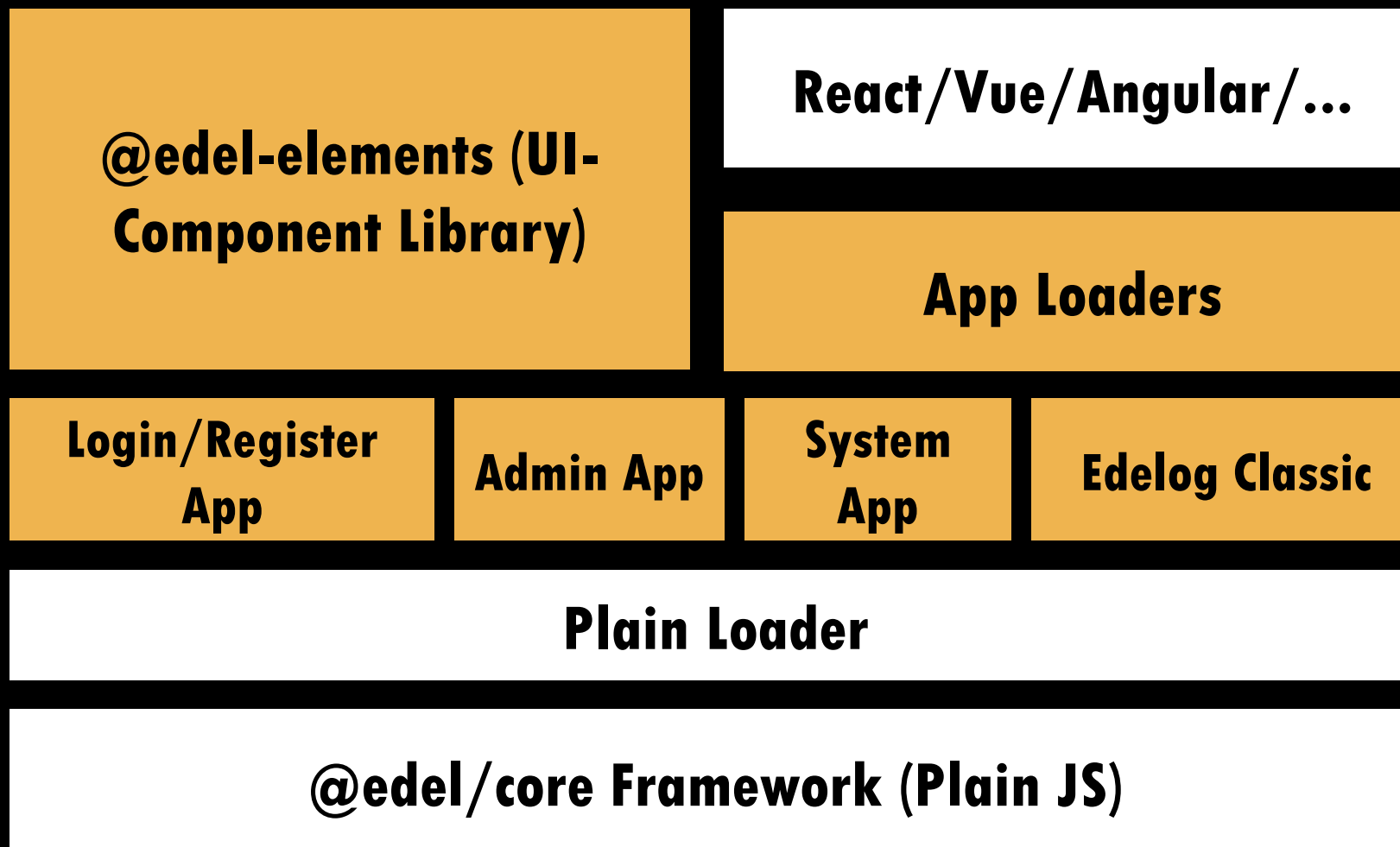
React/Vue/Angular/...

App Loaders

VueJS App

@edel/core Framework (Plain JS)

How we use it



Edelog & Edel-Elements

- www.edelog.com
- Publishing Edelog and Edel-Elements Open Source @ End 2019

Thank you!

 @Inoverse

 online@aaronczichon.de