

Web Components:



Natif



StencilJS



LitElement



Julien Renaux

GDE Web, Freelancer

@julienrenaux

<https://julienrenaux.fr/>



Toptal



Igor Laborie

Expert Web & Java

@ilaborie

igor@monkeypatch.io

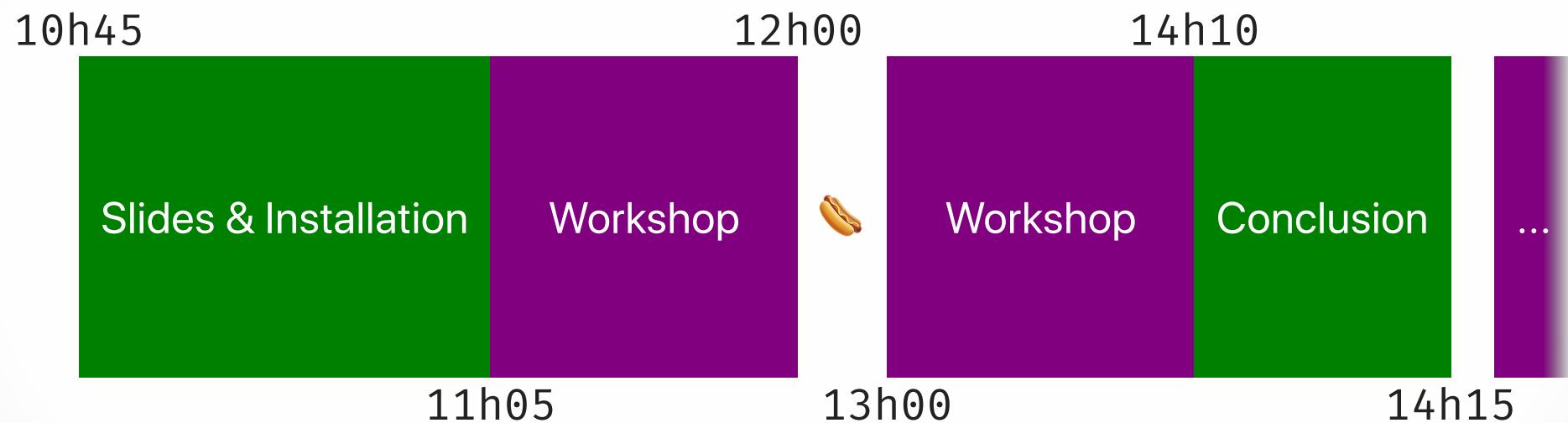


MonkeyPatch



Roadmap

#3



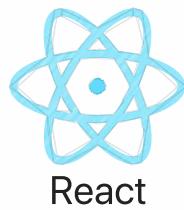
Instructions

#4

TODO: Installation instructions, Wifi

I want to build a Web app in 2019

Let's start by picking up a Framework



Now let's select how to write our style

- CSS
- Sass/Scss
- Less
- Stylus
- CSS-in-JS
- PostCSS
- NextCSS
- ...

Now let's transpile our code

- Webpack
- ParcelJs
- RollupJs
- Bazel

**Developing an app in JS is not easy
anymore...**

The industry is moving too fast...

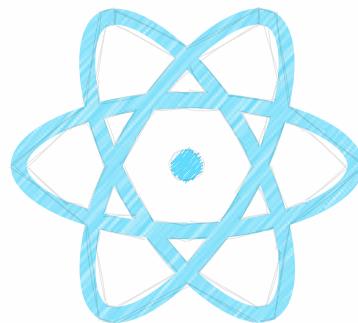
I Am Developer (@iamdevloper) posted a tweet on December 4, 2014, at 1:22 PM. The tweet reads: "I think I've had milk last longer than some JavaScript frameworks." It has received 1,129 likes and 1,755 people are talking about it. The Twitter logo is visible in the top right corner of the card.

I think I've had milk last longer than some JavaScript frameworks.

1,129 1:22 PM - Dec 4, 2014

1,755 people are talking about this >

**...Interoperability is not available out
of the box...**



React

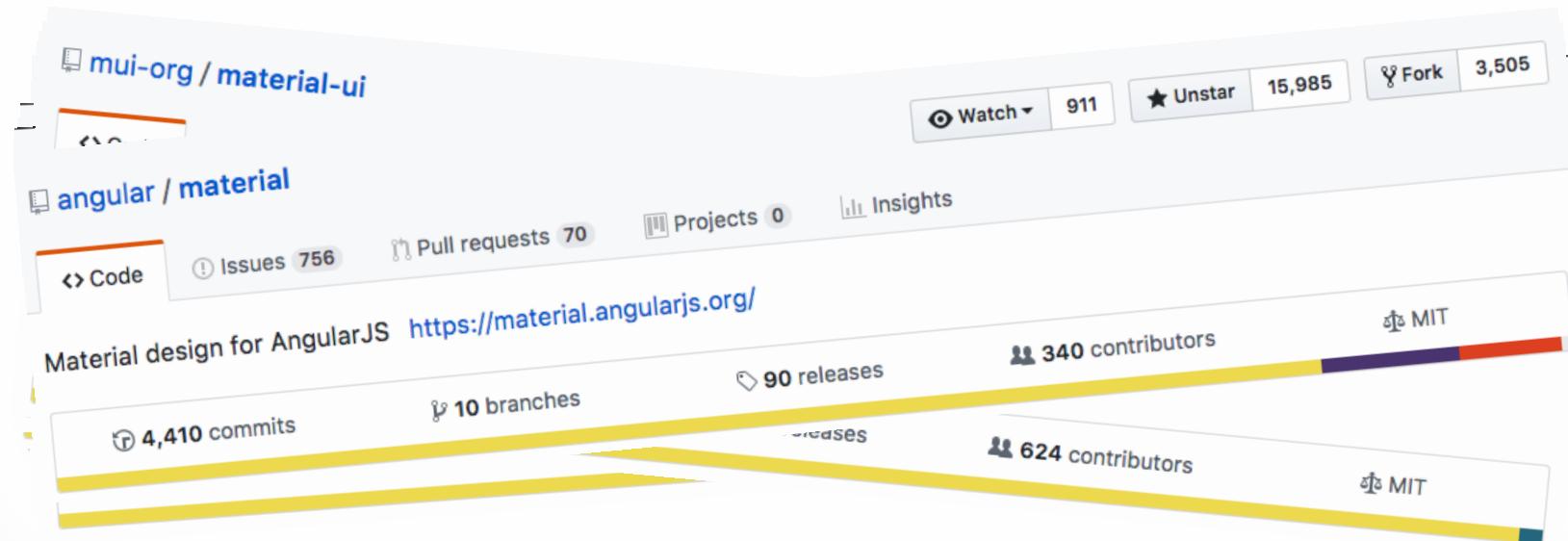


Vue.js

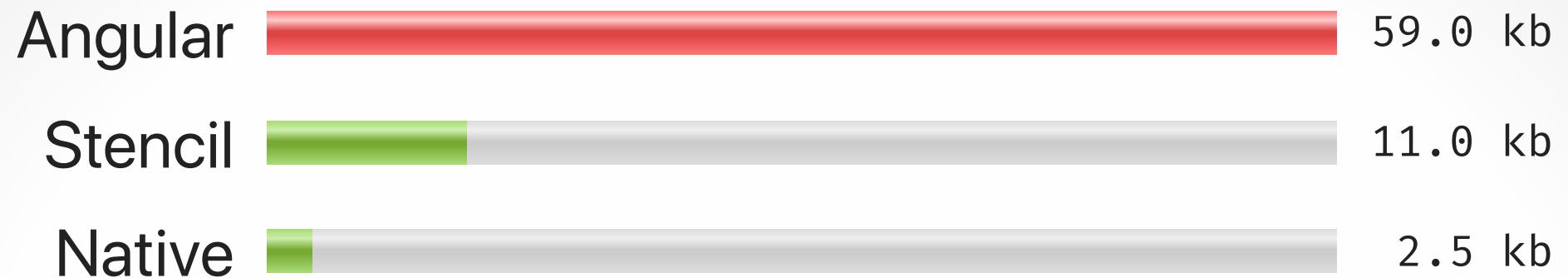


Angular

...And we keep reinventing the wheel!



**All this complexity is coming to an
end with Web Components**

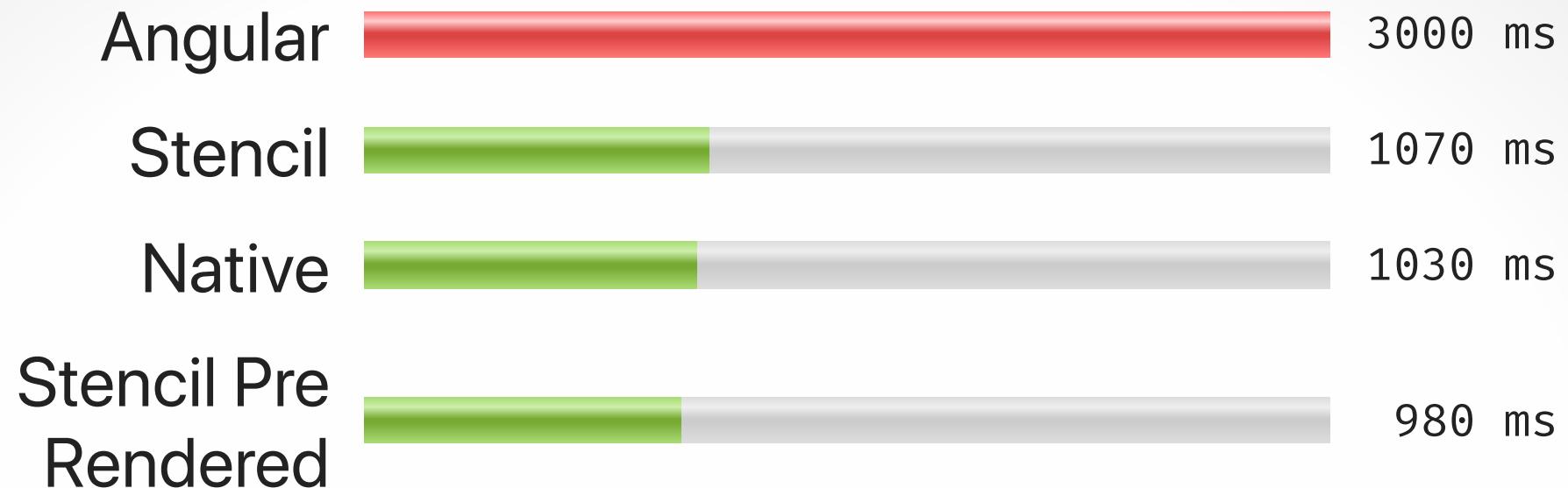


Size matters (Gzipped)

怃 Stencil is 5 times smaller than Angular

;o Native is 23 times smaller than Angular

TODO: Add litElement, Update numbers



Time matters (FMP 3G 📱 in ms)

😱 Native & Stencil 3 times faster than Angular

TODO: Add litElement, Update numbers

Web Components



Specs from **World Wide Web Consortium** (W3C)
First draft in **2012**



Custom
Elements



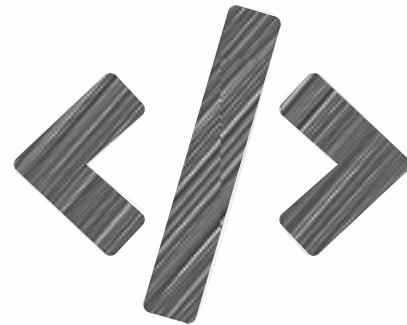
Shadow DOM



HTML
templates



HTML
imports



Custom Elements

“Custom Elements is a capability for creating your own custom HTML elements with its own methods and properties”



Shadow DOM

“ *Shadow DOM provides encapsulation for DOM and CSS* ”



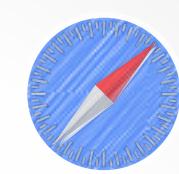
HTML templates

“ Give the ability to create reusable piece of HTML that can be used at runtime ”



```
1 class PopUpInfo extends HTMLElement {  
2     constructor() {  
3         super();  
4         // ...  
5     }  
6     // ...  
7 }  
8  
9 customElements.define('popup-info', PopUpInfo);
```

Browser support

	 11	 18	 65	 72	 12
Custom Elements (V1)					
Shadow DOM (V1)					
HTML templates					

Supported

Partial Support

Not Supported

<https://caniuse.com>

Polyfill support



11+

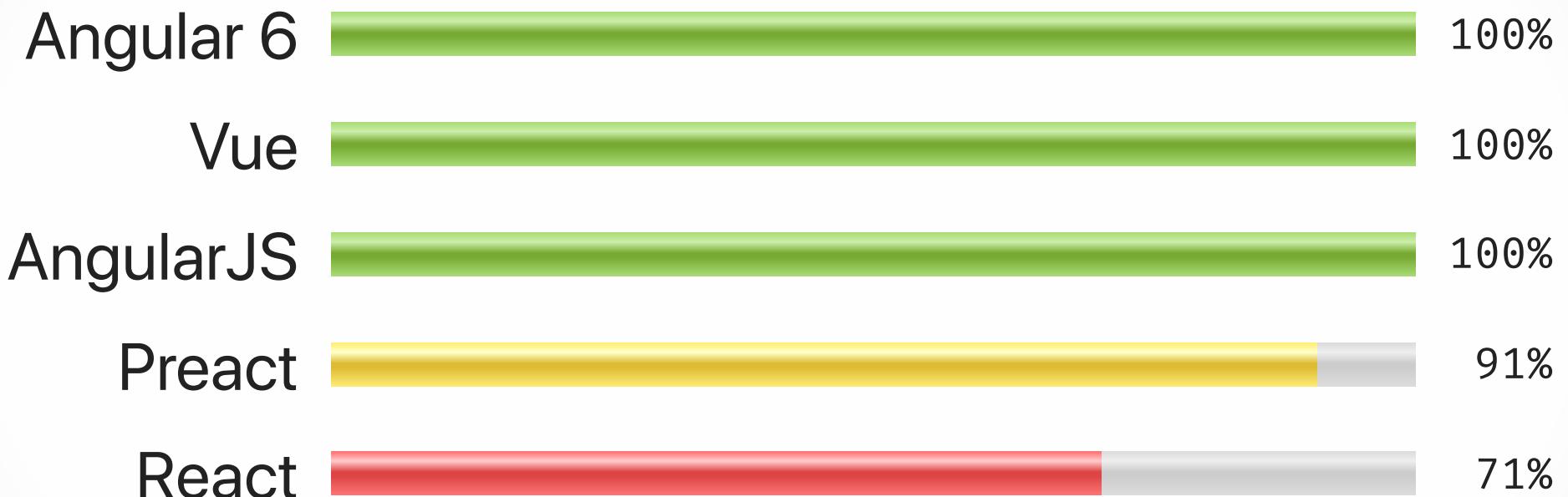


9+

	IE 11+	Edge	Firefox	Chrome	Safari 9+
Custom Elements (V1)	✓	✓	✓	✓	✓
Shadow DOM (V1)	✓	✓	✓	✓	✓
HTML templates	✓	✓	✓	✓	✓

<https://github.com/webcomponents/webcomponentsjs>

Web component interoperability support



<https://custom-elements-everywhere.com/>

StencilJS



<https://stenciljs.com/>

Open Source project, ➔ MIT License

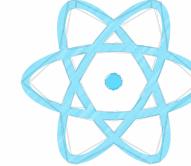
Created by the **Ionic Team** in 2017

4.9k  on github

StencilJS is not another framework

StencilJS is a **compiler that
generates **web components****

StencilJS is a **set** of great tools

				
JSX / Virtual DOM	✗	✗	✓	✓
TypeScript	✗	✓	✗	✓
Decorators	✗	✓	✗	✓
Prerendering SSR	✗	✗	✗	✓

StencilJS works everywhere

Loads polyfills on-demand

Stencil Syntax is concise

Web component

```

1 class TodoItem extends HTMLElement {
2   constructor() {
3     super();
4     this._root = this.attachShadow({ mode: 'open' });
5     this._checked = false;
6     this._text = '';
7   }
8   connectedCallback() {
9     this._root.innerHTML =
10    `
11    <li class="item">
12      <input type="checkbox">
13      <label>${this._text}</label>
14      <button class="destroy">x</button>
15    </li>`;
16    this.$item = this._root.querySelector('.item');
17    this.$removeButton = this._root.querySelector('.destroy');
18    this.$text = this._root.querySelector('label');
19    this.$checkbox = this._root.querySelector('input');
20    this.$removeButton.addEventListener('click', (e) => {
21      e.preventDefault();
22      this.dispatchEvent(new CustomEvent('onRemove', { detail: this.index }));
23    });
24    this.$checkbox.addEventListener('click', (e) => {
25      e.preventDefault();
26      this.dispatchEvent(new CustomEvent('onToggle', { detail: this.index }));
27    });
28  }
29  disconnectedCallback() {
30  }
31  static get observedAttributes() {
32    return ['text'];
33  }
34  attributeChangedCallback(name, oldValue, newValue) {
35    this._text = newValue;
36  }
37  set index(value) {
38    this._index = value;
39  }
40  get index() {
41    return this._index;
42  }
43  set checked(value) {
44    this._checked = Boolean(value);
45  }
46  get checked() {
47    return this.hasAttribute('checked');
48  }
49  _render() {
50    if (!this.$item) return;
51    this.$text.textContent = this._text;
52    if (this._checked) {
53      this.$item.classList.add('completed');
54      this.$checkbox.setAttribute('checked', '');
55    } else {
56      this.$item.classList.remove('completed');
57      this.$checkbox.removeAttribute('checked');
58    }
59  }
60  window.customElements.define('todo-item', TodoItem);
61 }
62 
```

Polymer 2

```

1 <link rel="import" href="../../bower_components/polymer/polymer-element.html">
2 <dom-module id="todo-item">
3   <template>
4     <style>...
5     </style>
6     <li class="item" [isCompleted="checked"]>
7       <input type="checkbox" value="checked" checked="checked" (change)=>handleOnChecked()
8       <label>${text}</label>
9       <button class="destroy" on-click="handleOnRemove">x</button>
10    </li>
11  </template>
12  <script>
13    class TodoItem extends Polymer.Element {
14      static get is() { return 'todo-item'; }
15      static get properties() {
16        return {
17          checked: { type: Boolean, value: false },
18          index: { type: Number, },
19          text: { type: String, value: '' }
20        };
21      }
22      handleOnRemove(e) {
23        this.dispatchEvent(new CustomEvent('remove', { detail: this.index }));
24      }
25      handleOnCheck(e) {
26        this.dispatchEvent(new CustomEvent('toggle', { detail: this.index }));
27      }
28      isCompleted(completed) {
29        return completed ? 'completed' : '';
30      }
31    }
32    window.customElements.define(TodoItem.is, TodoItem);
33  </script>
34 
```

Angular Elements

```

1 import { Component, EventEmitter, Input, Output, ViewEncapsulation } from '@angular/core';
2 @Component({
3   selector: 'todo-item',
4   template: `
5     <li class="item" [class.completed]="checked">
6       <input type="checkbox" [checked]="checked" (change)=>handleOnChecked()
7       <label>${text}</label>
8       <button class="destroy" (click)=>handleOnRemove(x)</button>
9     </li>
10    `,
11    styles: ['<!-- styles -->']
12  })
13  export class TodoItem {
14    @Input() checked: boolean;
15    @Input() text: string;
16    @Input() index: number;
17    @Event() onTodoItemChecked: EventEmitter;
18    @Event() onTodoItemRemove: EventEmitter;
19    handleOnRemove = () => this.onTodoItemRemove.emit(this.index);
20    handleOnChecked = () => this.onTodoItemChecked.emit(this.index);
21  }
22 
```

StencilJS

```

1 import { Component, Prop, Event, EventEmitter } from '@stencil/core';
2 @Component({
3   tag: 'todo-item',
4   styleUrl: 'todo-item.scss',
5   shadow: true,
6 })
7 export class TodoItem {
8   @Prop() checked: boolean;
9   @Prop() text: string;
10  @Prop() index: number;
11  @Event() onTodoItemChecked: EventEmitter;
12  @Event() onTodoItemRemove: EventEmitter;
13  handleOnRemove = () => this.onTodoItemRemove.emit(this.index);
14  handleOnChecked = () => this.onTodoItemChecked.emit(this.index);
15  render() {
16    return (
17      <li class={this.checked ? 'completed' : ''}>
18        <input type="checkbox" checked={this.checked} onChange={this.handleOnCheck}>
19        <label>${this.text}</label>
20        <button onClick={this.handleOnRemove}>x</button>
21      </li>
22    );
23  }
24 
```

SkateJS + Preact

```

1 // @jsx h
2 import { props } from "skatejs/dist/esnext";
3 import { h } from "preact";
4 import { Component } from "./util";
5 
6 export default class extends Component {
7   static events = ["check", "remove"];
8   static props = {
9     checked: props.boolean,
10    index: props.number
11  };
12 
13  handleCheck = e => {
14    this.onCheck({ index: this.index, value: e.target.checked });
15  };
16  handleRemove = () => {
17    this.onRemove({ index: this.index });
18  };
19 
20  render({ checked, handleCheck, handleRemove }) {
21    return (
22      <div>
23        <style>{<!--
24          <!-- styles -->
25        </style>
26        <li class={checked ? "completed" : ""}>
27          <input type="checkbox" checked={checked} onChange={handleCheck}> />
28          <label>${text}</label>
29          <slot></slot>
30          <button onClick={handleRemove}>x</button>
31        </li>
32      </div>
33    );
34  }
35 
```

SkateJS + lit-html

```

1 import { props } from "skatejs/dist/esnext";
2 import { html } from "lit-html/lib/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  handleRemove = () => {
16    this.onRemove({ index: this.index });
17  };
18 
19  render({ checked, handleCheck, handleRemove }) {
20    return html`<div>
21      <style>{<!--
22        <!-- styles -->
23      </style>
24      <li class={checked ? "completed" : ""}>
25        <input type="checkbox" checked="${checked}" on-change="${handleCheck}" />
26        <label>${text}</label>
27        <slot></slot>
28        <button on-click="${handleRemove}">x</button>
29      </li>
30    </div>
31  `;
32 
```

Getting started

#33



```
$ npm init stencil
```



```
? Pick a starter > - Use arrow-keys. Return to submit.  
> ionic-pwa      Everything you need to build fast, production ready PWAs  
    app           Minimal starter for building a Stencil app or website  
    component     Collection of web components that can be used anywhere
```



A screenshot of a code editor window showing a TypeScript file. The file contains a component definition for 'MyComponent'. The code includes imports from '@stencil/core', a component annotation with tag and style URL, an export class declaration, a prop named 'name' with a type of string, and a render method returning a JSX element. The code editor has a dark theme, red, yellow, and green window controls at the top left, and a 'TS' status indicator at the top right.

```
1 import {Component, Prop} from '@stencil/core';
2
3 @Component({
4   tag: 'my-first-component',
5   styleUrl: 'my-first-component.scss'
6 })
7 export class MyComponent {
8   // Indicate that name should be
9   // a public property on the component
10  @Prop() name: string;
11
12  render() {
13    // JSX
14    return (<p>My name is {this.name}</p>);
15  }
16}
```

Lit-Elements



<https://lit-element.polymer-project.org/>

Open Source project, ➔ BSD 3-Clause License

Created by the **Polymer Team** in 2017

1.7k  on github

Close to other lightweight WebComponent frameworks

- ➡ **Stencil**
- ➡ **SkateJS**
- ➡ **Svelte**
- ➡ **Slim.js**

But using the lit-html templating library

<https://lit-html.polymer-project.org/>

built on **HTML templates**
with the ES2015 ➔ **Template literals**



JS

```
1 import {html, render} from 'lit-html';
2
3 // A lit-html template uses
4 // the `html` template tag:
5 const sayHello = (name) =>
6   html`<h1>Hello ${name}</h1>`;
7
8 // It's rendered with the `render()` function:
9 render(sayHello('World'), document.body);
10
11 // And re-renders only update the
12 // data that changed, without VDOM diffing!
13 render(sayHello('Everyone'), document.body);
```

Just extend the LitElement class

```
1 import { LitElement, html, css } from 'https://unpkg.com/lit-element/l
2
3 class MyElement extends LitElement {
4     static get properties() {
5         return {
6             mood: { type: String }
7         }
8     }
9     static get styles() {
10        return css `.
11        .mood { color: green; }`;
12    }
13    render() {
14        return html `Web Components are
15            <span class="mood">${this.mood}</span>!;
16    }
17
18 customElements.define('my-element', MyElement);
```

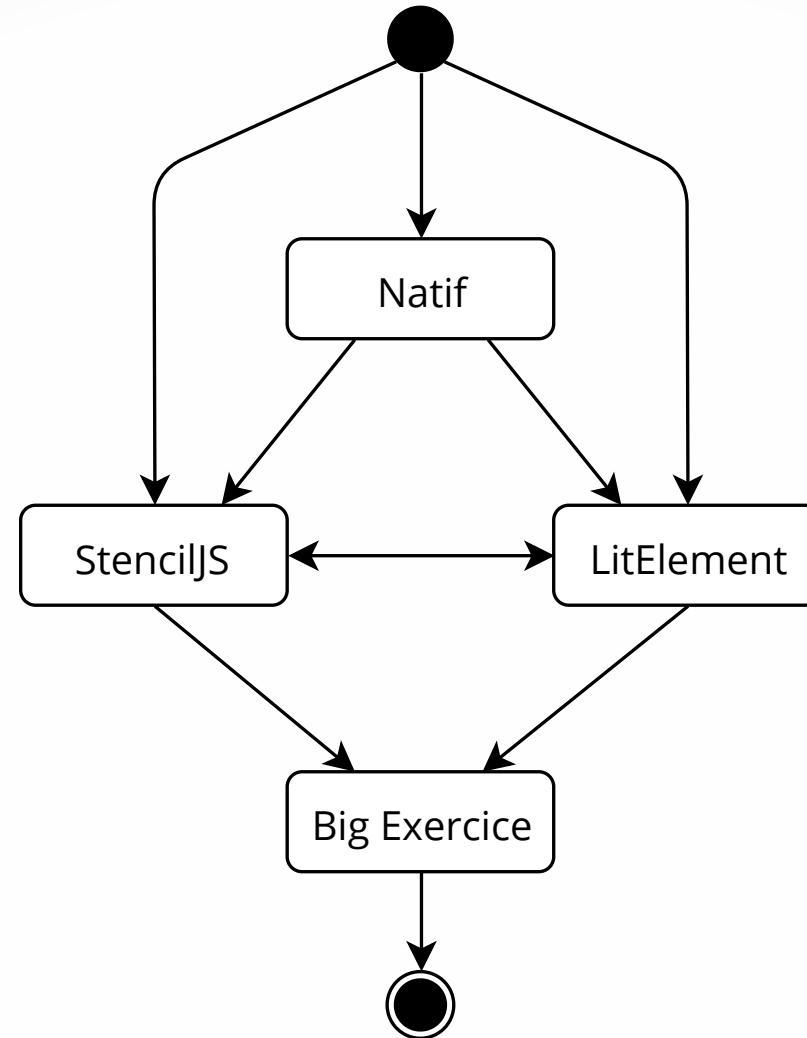
Also can use Typescript with **decorators**

```
 1 import { /* ... */ } from 'lit-element';
 2
 3 @customElement('my-element')
 4 class MyElement extends LitElement {
 5
 6     @property({ type: String }) mood;
 7
 8     static get styles() {
 9         return css`mood { color: green; }`;
10     }
11
12     render() {
13         return html`Web Components are
14             <span class="mood">${this.mood}</span>`;
15     }
16 }
```

 Chrome, Safari, Opera, Firefox

 polyfills for Edge and IE 11

Workshop



Conclusion

Commons Issues

#46

Attribute are `string`

Use an external state manager

Styling with theme

Use CSS custom properties

Browser support

Polyfills or Electron

TODO: ...



Moderns Alternatives

#47

- ➡ SkateJS
- ➡ Svelte
- ➡ Slim.js

...

Thanks