

Web Components:



Natif



StencilJS



LitElement





Julien Renaux

GDE Web, Freelancer

@julienrenaux

<https://julienrenaux.fr/>



Igor Laborie

Expert Web & Java

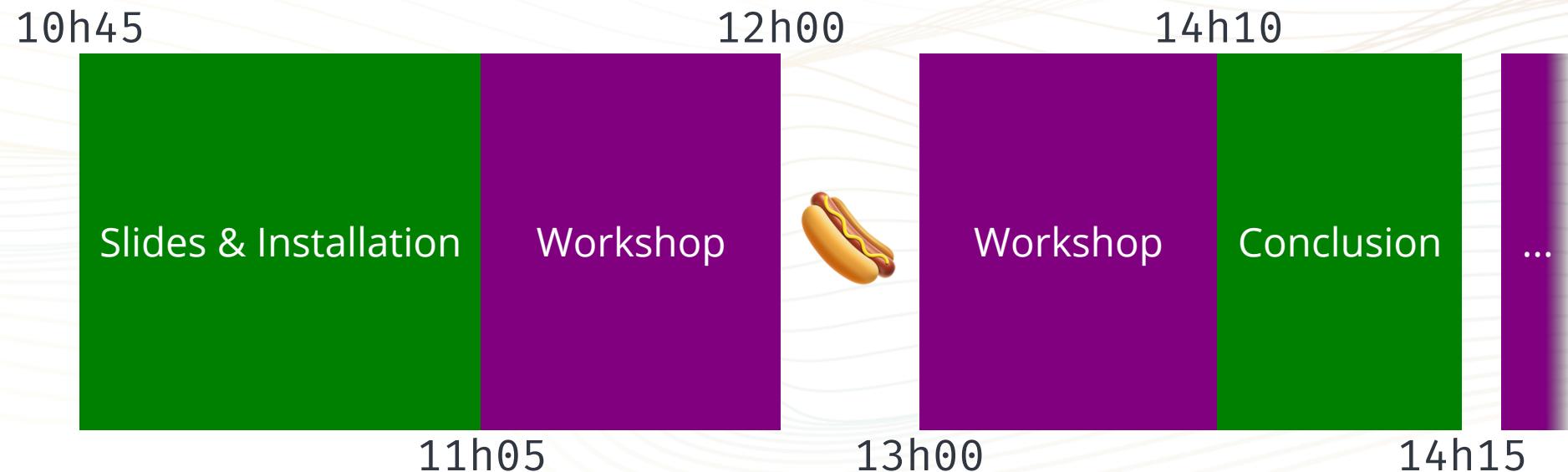
@ilaborie

igor@monkeypatch.io





Roadmap

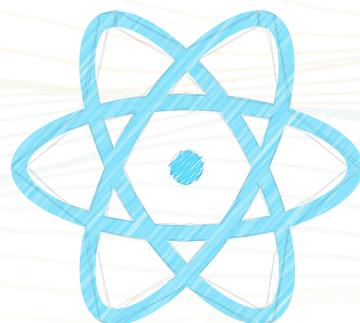


Instructions

TODO: Installation instructions, Wifi

Développons une application Web en 2019

Commençons par choisir un Framework



React



Vue.js



Angular

Puis comment nous allons écrire le styles

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

Puis construisons notre application

- Webpack
- ParcelJs
- RollupJs
- Bazel



**Développer une application en JS
n'est plus simple...**

Ça va trop vite ...

A screenshot of a Twitter post. The profile picture is a small circular image of a person with short hair. The username is "I Am Devloper" and the handle is "@jamdevloper". The tweet text is: "I think I've had milk last longer than some JavaScript frameworks." Below the tweet are engagement metrics: 1,125 likes and 1,755 people talking about it. There are also icons for a blue bird (Twitter logo), a magnifying glass (info), and a right-pointing arrow.

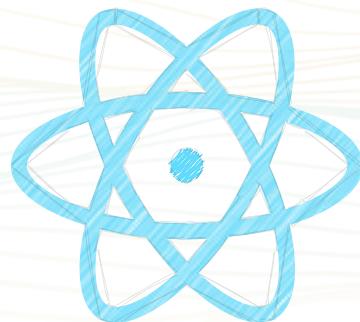
I Am Devloper
@jamdevloper

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

1,125 1:22 PM - Dec 4, 2014

1,755 people are talking about this >

...Interopérabilité ne vient plus gratuitement...



React

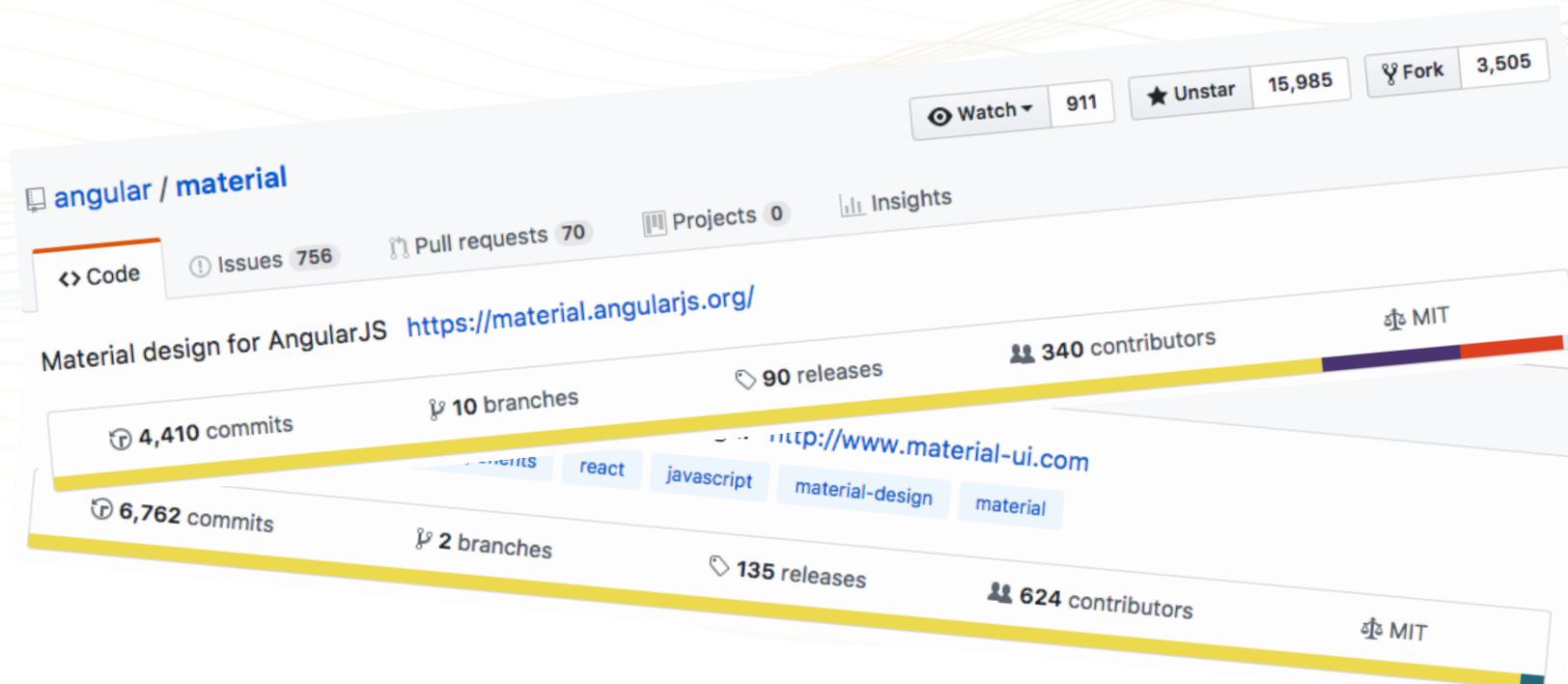


Vue.js



Angular

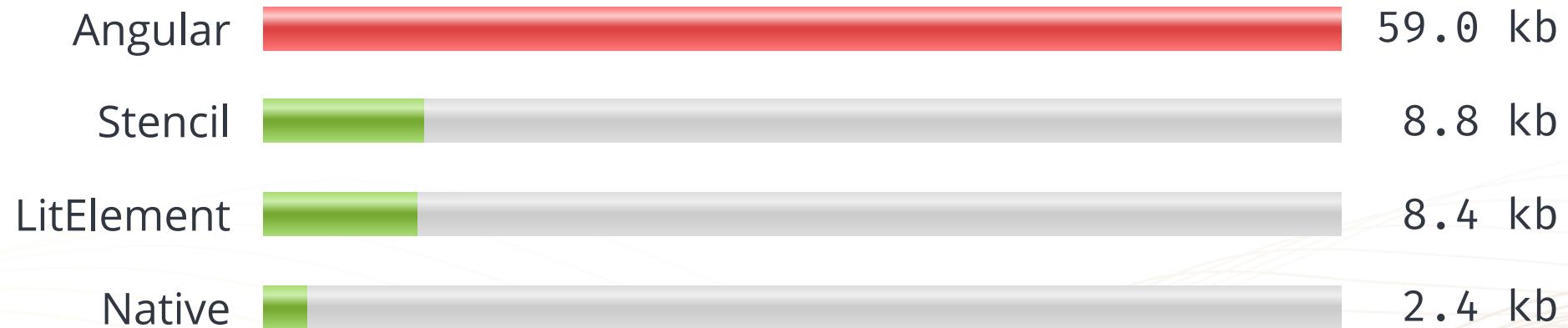
...et on réinvente sans arrêt la roue !



TODO: Meme Luci Do It: add another one in
WebComponent LitElement...

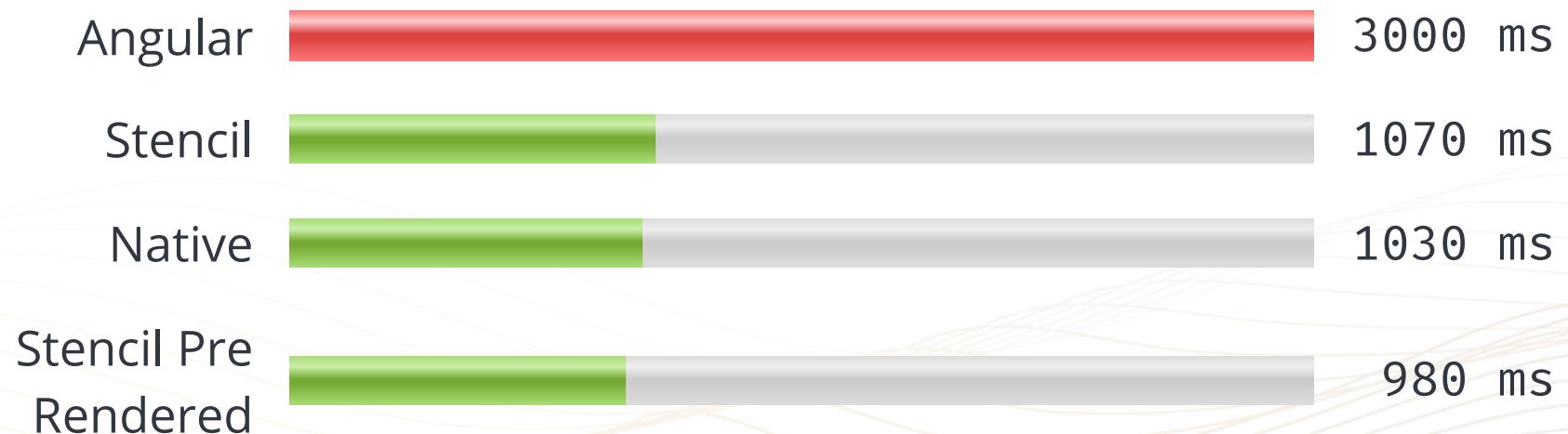


**Toute cette complexité doit
s'arrêter avec les Web
Components**



Size matters (Gzipped)

- 😢 Stencil et litElement sont 5 fois plus petit qu'Angular
- 😱 Native est 23 fois plus petit qu'Angular



Le temps ça compte (FMP 3G 📱 en ms)

😢 Native et Stencil sont 3 fois plus rapide qu'Angular

Web Components



Spécifier par le **World Wide Web Consortium**
(W3C)

Débuté en **2012**



Custom
Elements



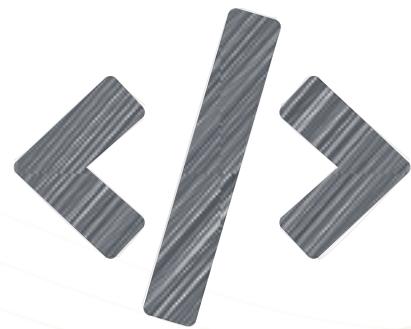
Shadow DOM



HTML
templates



HTML
Imports



Custom Elements

“ Les Custom Elements sont la capacité de créer une propres balise HTML avec ses propres attributs et méthodes



Shadow DOM

**// Le Shadow DOM fournit l'encapsulation du
DOM et du CSS**



HTML templates

//

Définit un bloc d'HTML réutilisable au moment de l'exécution

A screenshot of a code editor window. The title bar has three colored circles (red, yellow, green) on the left and a yellow 'JS' icon on the right. The main area contains the following code:

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

Support des navigateurs

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

Supported

Partial Support

Not Supported

<https://caniuse.com>

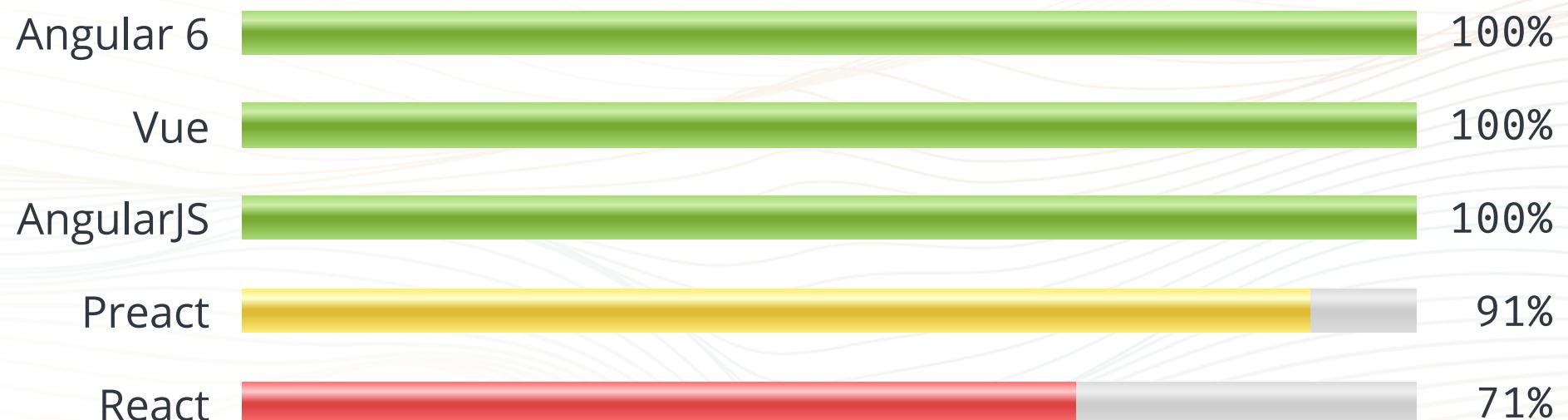
Support avec le polyfill



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

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

Interopérabilité des Web Component



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

StencilJS



<https://stenciljs.com/>



@julienrenaux @ilaborie #DevoxxFR #webcomponents

DEVOXX
France

Projet **Open Source**, ➔ MIT License

Créer par l'équipe d'**Ionic** en 2017

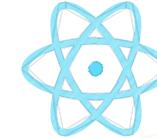
5.1k ⭐ sur github



**StencilJS n'est pas un autre
framework**

**StencilJS c'est un compilateur qui
génère des web components**

StencilJS c'est un ensemble de bons outils

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

StencilJS marche partout

Il charge les polyfills à la demande

La syntax de Stencil est concise

Web component	Polymer 2	Angular Elements	StencilJS
<pre>1 class TodoItem extends HTMLElement { 2 constructor() { 3 super(); 4 this._root = this.attachShadow({ 'mode': 'open' }); 5 this._text = ''; 6 } 7 connectedCallback() { 8 this._root.innerHTML = 9 `<style></style></pre>			

Pour démarrer



```
$ npm init stencil
```



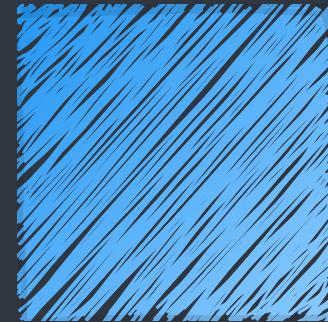
```
? Pick a starter > - Use arrow-keys. Return to submit.  
> ionic-pwa      Everything you need to build fast, production ready PWAs  
    app            Minimal starter for building a Stencil app or website  
    component     Collection of web components that can be used anywhere
```



A screenshot of a code editor window showing a TypeScript file. The file contains a component definition named MyComponent. The code includes imports for Component and Prop from @stencil/core, a component annotation with tag 'my-first-component' and styleUrl 'my-first-component.scss', and a render method returning a JSX element with the name prop. The code editor has a dark theme with syntax highlighting and a status bar indicating 'TS'.

```
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/>



Projet **Open Source**, ➔ BSD 3-Clause License

Créer par l'équipe **Polymer Team** en 2017

1.7k ★ sur github

Utilise la bibliothèque de template **lit-html**

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

Basé sur les **templates HTML**

Avec les ➔ *Template literals* de ES2015



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);
```

 JS

```
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`mood { color: green; }`;
11    }
12    render() {
13        return html `Web Components are
14            <span class="mood">${this.mood}</span>!`;
15    }
16}
17
18 customElements.define('my-element', MyElement);
```



```
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}
```

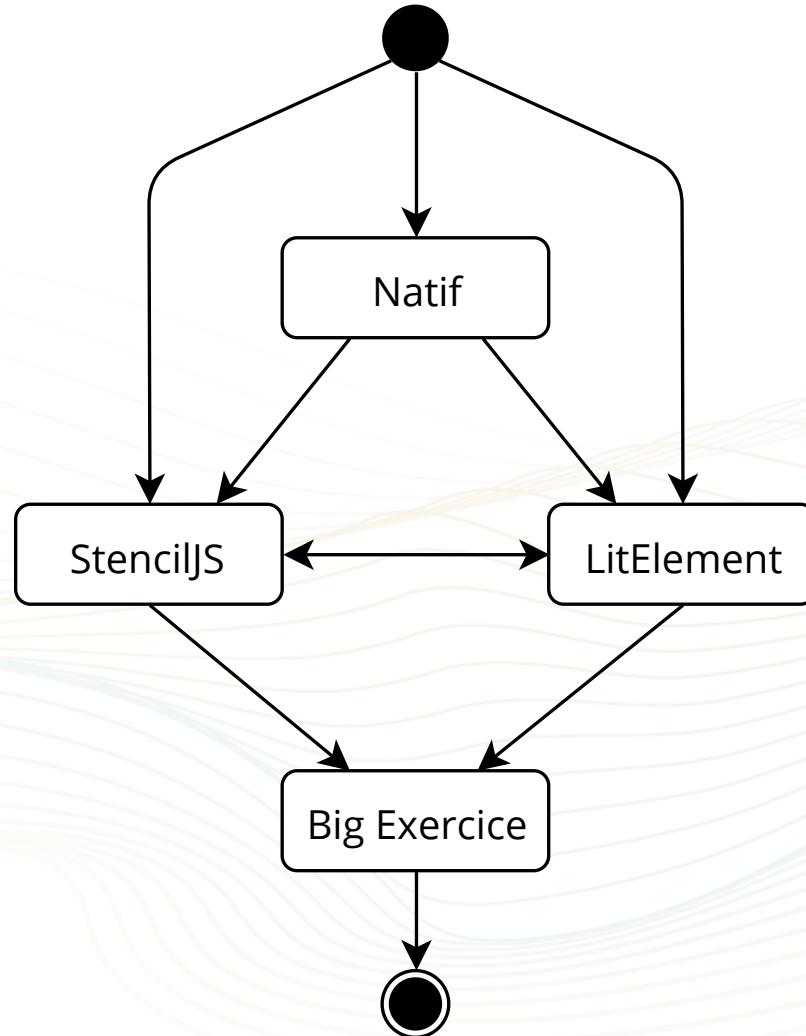
 Chrome, Safari, Opera, Firefox

 polyfills pour Edge et IE 11

Workshop

@julienrenaux @ilaborie #DevoxxFR #webcomponents

DEVOXX
France



Conclusion

@julienrenaux @ilaborie #DevoxxFR #webcomponents

DEVOXX
France

Construire une
Application

Style avec un thème

Support des navigateurs Utiliser le polyfill ou
Electron

TODO: ...

Alternatives modernes

➤ SkateJS

➤ Svelte

➤ Slim.js

...

Fin

Merci

Pensez à nous faire des retours (votez !)