Web Components

with **Angular**

Hello!
I’m Sherry List

Azure Developer Technical Lead, Microsoft
Women Techmaker Lead

You can find me at @SherrryLst





Hi!
I'm Ana Cidre

Developer Advocate, Ultimate Courses
Women Techmaker Lead

You can find me at @AnaCidre_





Web Components

A component that is **platform agnostic**

Their main goal is to **encapsulate** the code for the components into a nice, **reusable** package for maximum **interoperability**.

Custom Elements Everywhere

Making sure frameworks and custom elements can be BFFs 

Check out: **https://custom-elements-everywhere.com/**

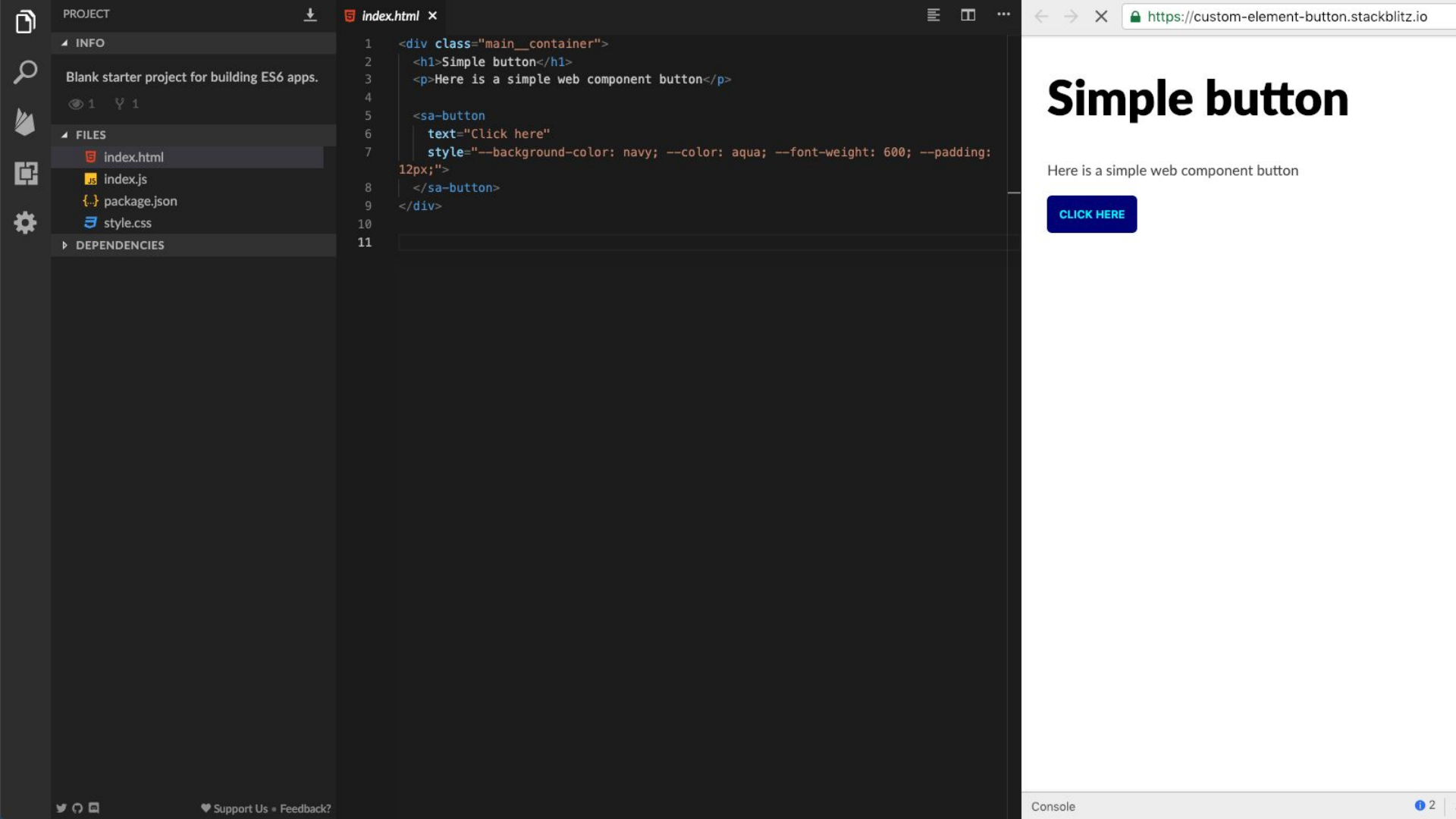
Related Issues

Yay! No open issues!

LIBRARY	SCORE	BASIC TESTS	ADVANCED TESTS
Angular 7.2.0	100%	16/16	14/14

Web components consist of three main technologies:

- HTML template
- Custom Elements
- Shadow DOM
- ~~HTML imports~~


PROJECT
INFO
Blank starter project for building ES6 apps.
1 1
FILES
index.html
index.js
package.json
style.css
DEPENDENCIES
Support Us • Feedback?
index.html
1
2 <h1>Simple button</h1>
3 Here is a simple web component button
4
5 <sa-button
6 text="Click here"
7 style="--background-color: navy; --color: aqua; --font-weight: 600; --padding: 12px;">
8 </sa-button>
9
10
11
https://custom-element-button.stackblitz.io
Simple button
Here is a simple web component button
CLICK HERE
Console
2

html

```
<sa-button
    text="Click here"
    style="--background-color: navy; --color: aqua; --font-weight: 600; --padding: 12px;">
</sa-button>
```

js

```
const template = `
  <style>
   .btn {
    font-weight: var(--font-weight, 200);
    background-color: var(--background-color, indianred);
    border: 0;
    border-radius: 5px;
    color: var(--color, #fff);
    padding: var(--padding, 10px);
    text-transform: uppercase;
   }
  </style>
    <button id="text" class="btn"></button>
     `;
```

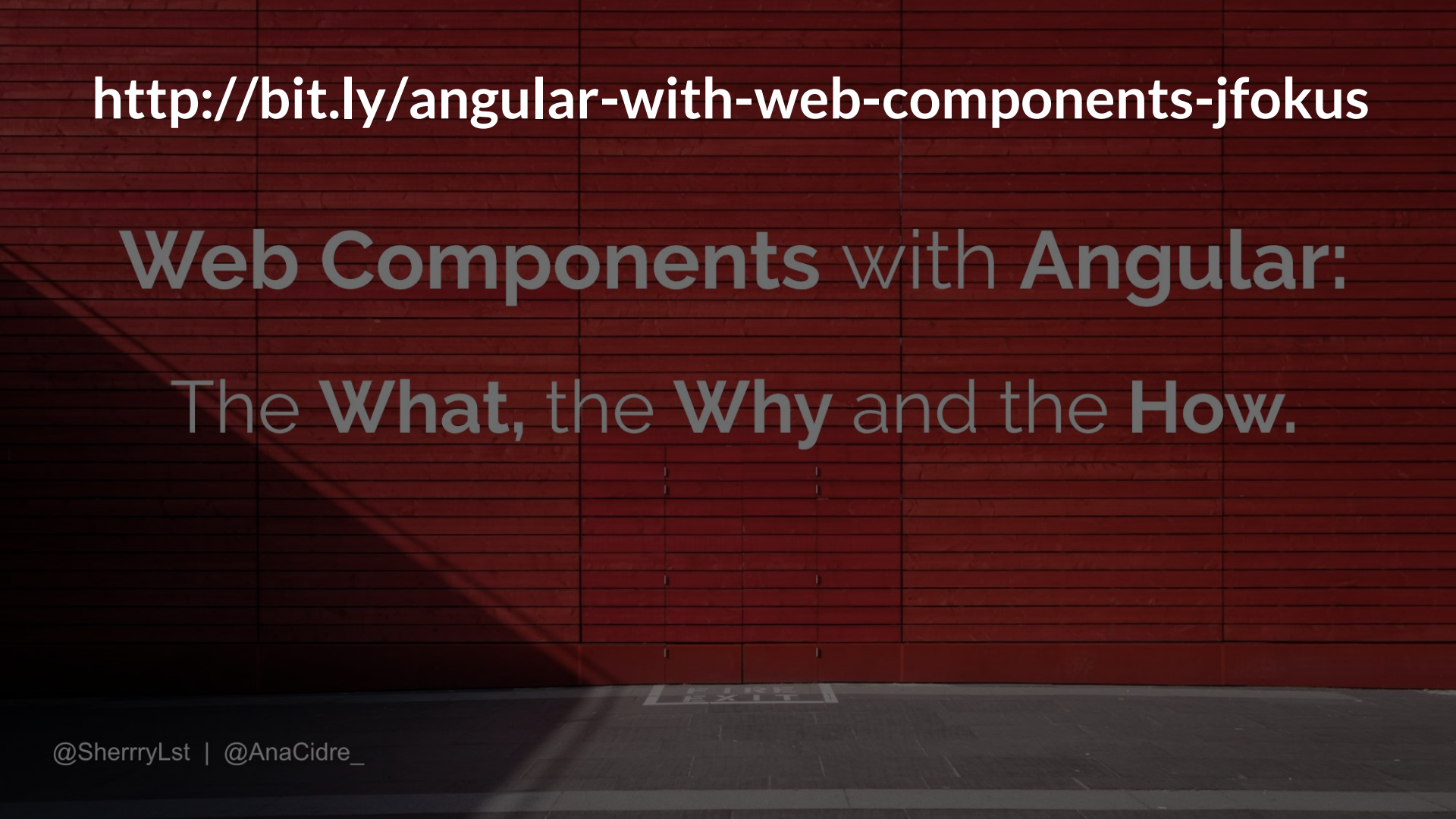
js

```
class SAButtonElement extends HTMLElement {
 constructor() {
  super();
  const template = document.createElement("template");

  // Shadow DOM
  this.attachShadow({ "mode": "open" });
  this.shadowRoot.appendChild(template.content.cloneNode(true));
}
```

js

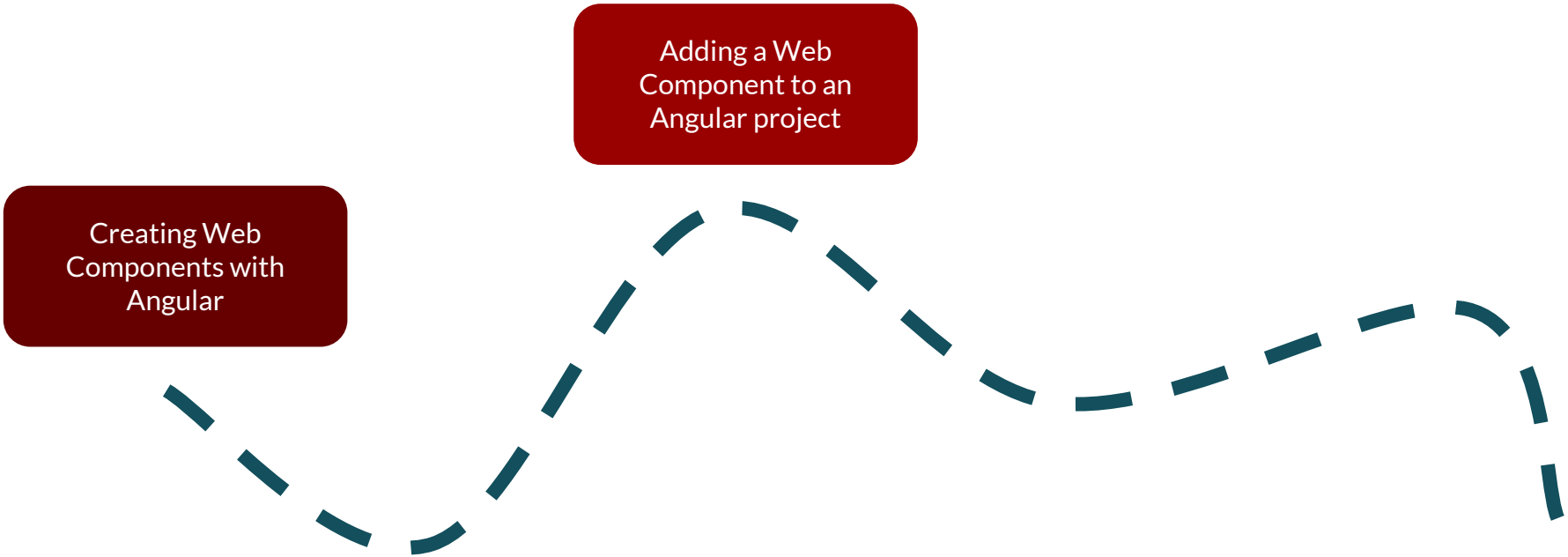
```
class SAButtonElement extends HTMLElement {
 constructor() {
  super();
  const template = document.createElement("template");

  // Shadow DOM
  this.attachShadow({ "mode": "open" });
  this.shadowRoot.appendChild(template.content.cloneNode(true));
 }
...
}
// Define custom element
customElements.define("sa-button", SAButtonElement);
```

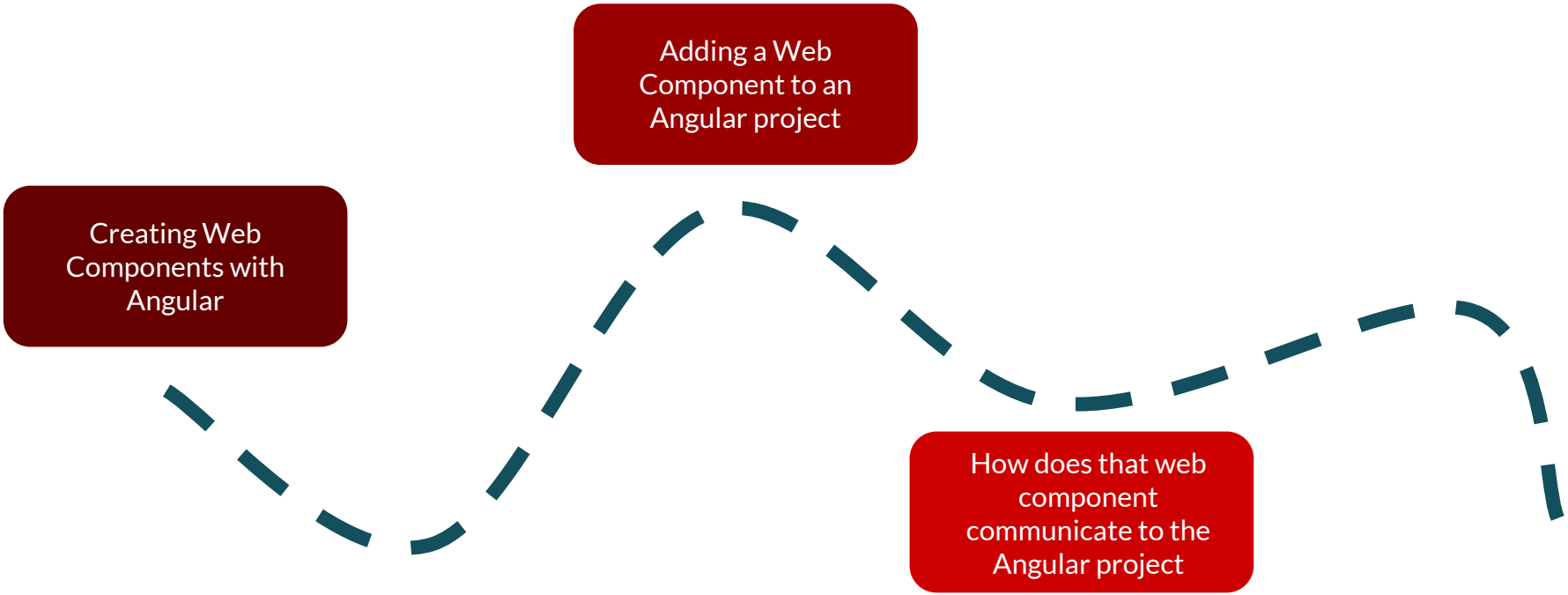
http://bit.ly/angular-with-web-components-jfokus

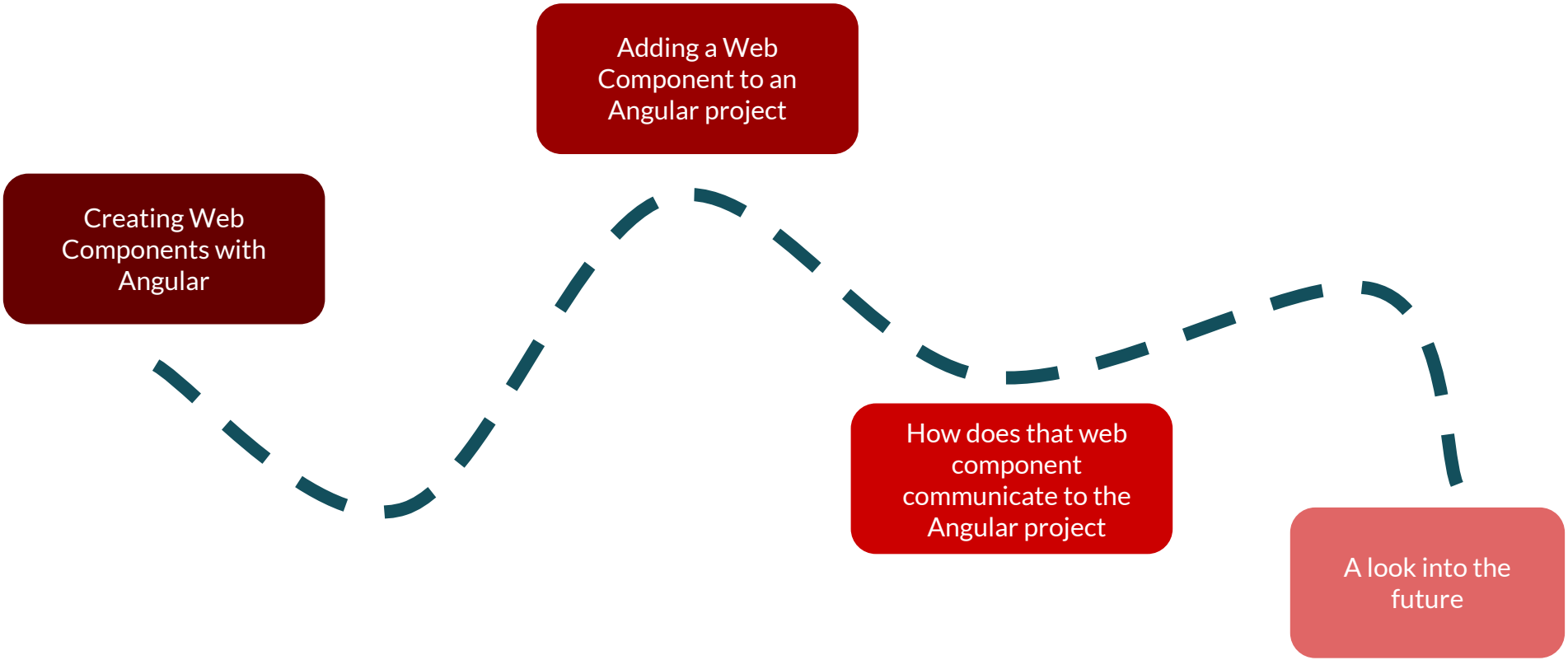


@SherrryLst | @AnaCidre_

Creating Web Components with Angular


Adding a Web Component to an Angular project
Creating Web Components with Angular


Creating Web Components with Angular
Adding a Web Component to an Angular project
How does that web component communicate to the Angular project


Creating Web Components with Angular
Adding a Web Component to an Angular project
How does that web component communicate to the Angular project
A look into the future

Angular Elements

terminal

```
npm i -g @angular/cli
```

terminal

```
npm i -g @angular/cli
```

```
ng new sa-button --prefix sa --inline-template --style=scss
```

terminal

```
npm i -g @angular/cli

ng new sa-button --prefix sa --inline-template --style=scss

cd sa-button
```

terminal

```
npm i -g @angular/cli

ng new sa-button --prefix sa --inline-template --style=scss

cd sa-button

ng add @angular/elements
```

terminal

```
npm i -g @angular/cli

ng new sa-button --prefix sa --inline-template --style=scss

cd sa-button

ng add @angular/elements

npm install @webcomponents/custom-elements --save
```

polyfills.ts

```
import '@webcomponents/custom-elements/custom-elements.min';
```

tsconfig.json

```
{
    "compileOnSave": false,
    "compilerOptions": {
        ...
        "target": "es2015",
        ....
        }
    }
}
```

terminal

```
ng generate component button
```

button.component.ts

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'sa-button',
  template: ` ...`,
  styleUrls: ['./button.component.scss'],
})

export class ButtonComponent implements OnInit {
 constructor() { }
 ngOnInit() {}
}
```

button.component.ts

```
import { Component, OnInit, Input } from '@angular/core';

@Component({
  selector: 'sa-button',
  template: ` <button id="text" class="btn">{{ text }}</button>`,
  styleUrls: ['./button.component.scss'],
})

export class ButtonComponent implements OnInit {
 @Input() text= 'Click me!';
 constructor() { }
 ngOnInit() {}
}
```

Web components consist of three main technologies:

- HTML template
- **Shadow DOM**
- Custom Elements

ViewEncapsulation.**ShadowDom**

This encapsulation mode uses the Shadow DOM to **scope styles only to this specific component.**

button.component.ts

```
import { Component, OnInit, ViewEncapsulation, Input } from '@angular/core';

@Component({
  selector: 'sa-button',
  template: ` ... `,
  styleUrls: ['./'sa-button'.scss'],
   encapsulation: ViewEncapsulation.ShadowDom
})
```

Web components consist of three main technologies:

- HTML template
- Shadow DOM
- **Custom Elements**

app.module.ts

```
import { NgModule, Injector } from '@angular/core';
import { createCustomElement } from '@angular/elements';

@NgModule({
  declarations: [ ButtonComponent ],
  imports: [ BrowserModule ],
  entryComponents: [ ButtonComponent ]
})
```

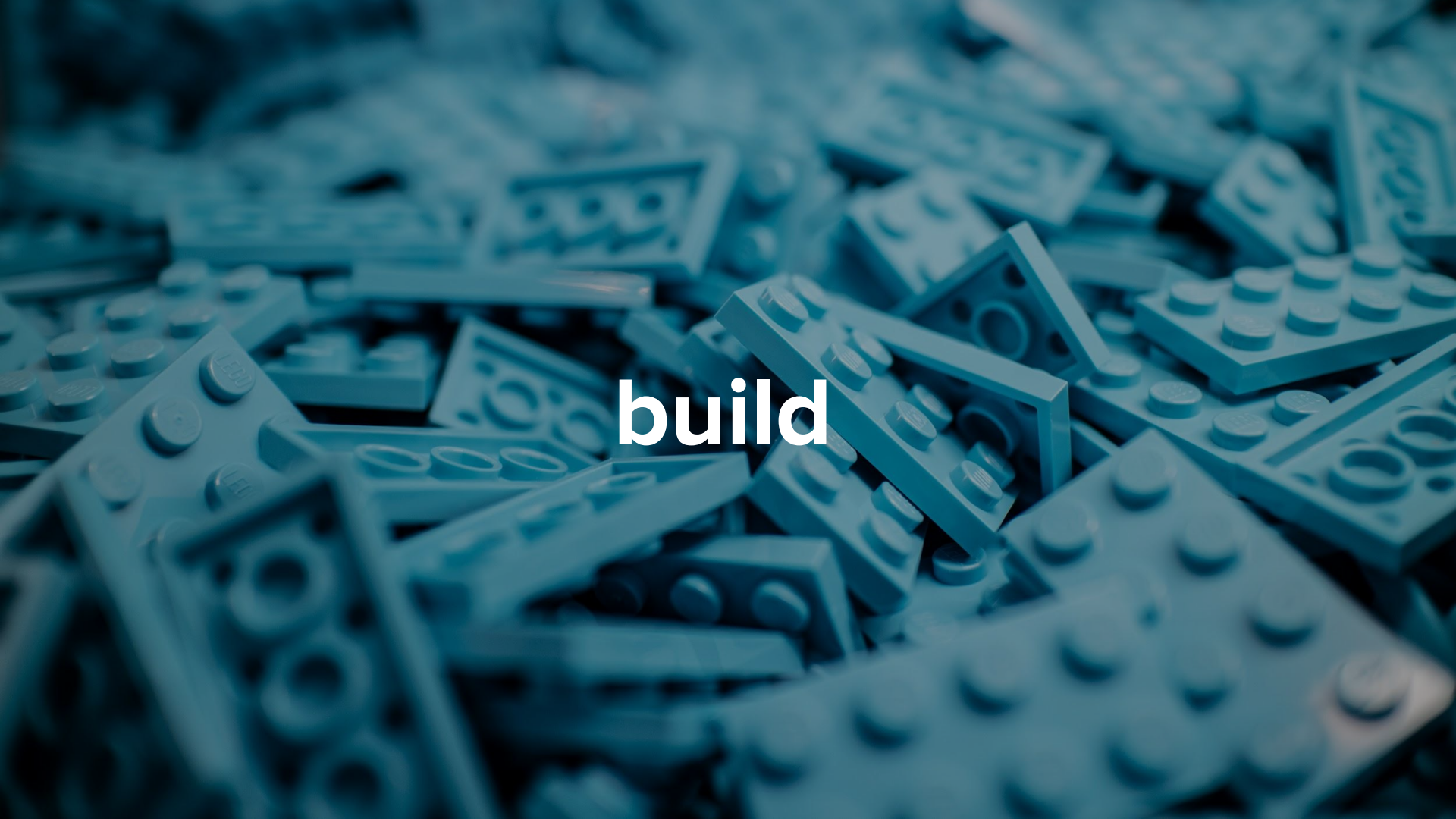
app.module.ts

```
@NgModule({
  declarations: [ButtonComponent],
  imports: [BrowserModule],
  entryComponents: [ButtonComponent],
})

export class AppModule {
 constructor(private injector: Injector) {
   const saButton = createCustomElement(ButtonComponent, { injector });
   customElements.define('sa-button', saButton);
 }
```

app.module.ts

```
@NgModule({
  declarations: [ButtonComponent],
  imports: [BrowserModule],
  entryComponents: [ButtonComponent],
})

export class AppModule {
 constructor(private injector: Injector) {
   const saButton = createCustomElement(ButtonComponent, { injector });
   customElements.define('sa-button', saButton);
 }
 ngDoBootstrap() {}
}
```

Wait a min


build

It’s not that easy!

- Huge bundle size
- No support for one bundle
- -> Eject
- Complicated!

ngx-build-plus

@ManfredSteyer

It’s that easy!

- Extends Cli
- No Eject
- Build a single bundle
- No need to add Angular multiple times to your project
- It’s simple!

How do we **implement** it?

terminal

```
ng add ngx-build-plus
```

angular.json

```
[...]
"architect": {
  "build": {
    "builder": "ngx-build-plus:build",
    [...]
  }
}
[...]
```

webpack.extra.js

```
module.exports = {
 externals: {
  rxjs: 'rxjs',
  '@angular/core': 'ng.core',
  '@angular/common': 'ng.common',
  '@angular/platform-browser': 'ng.platformBrowser',
  '@angular/elements': 'ng.elements',
 },
};
```

terminal

```
ng build --prod
         --extraWebpackConfig webpack.extra.js
         --single-bundle true
```

Now our component is **ready**!

Ivy

The new backwards-compatible Angular renderer:

- Speed improvements
- Bundle size reduction
- Increasing flexibility

How do we **use** our web component?

terminal

```
npm install @webcomponents/custom-elements  --save
```

polyfills.ts

```
import '@webcomponents/custom-elements/custom-elements.min';
```

app.module.ts

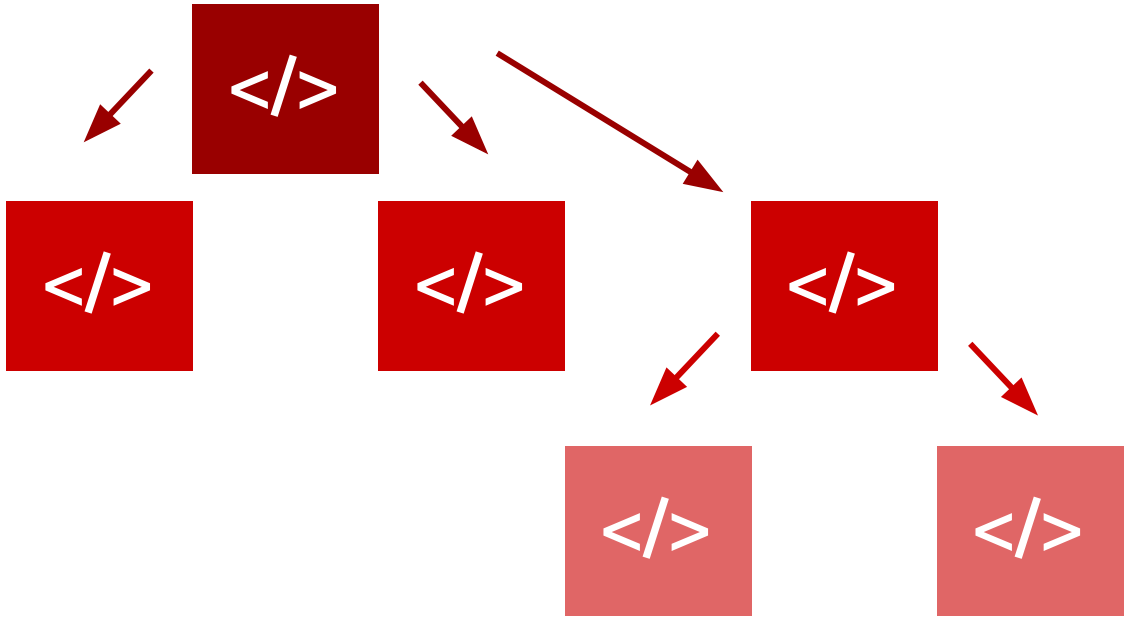
```
@NgModule({
  [...],
  schemas: [
    CUSTOM_ELEMENTS_SCHEMA
  ]
})


export class AppModule { }
```

app.component.ts

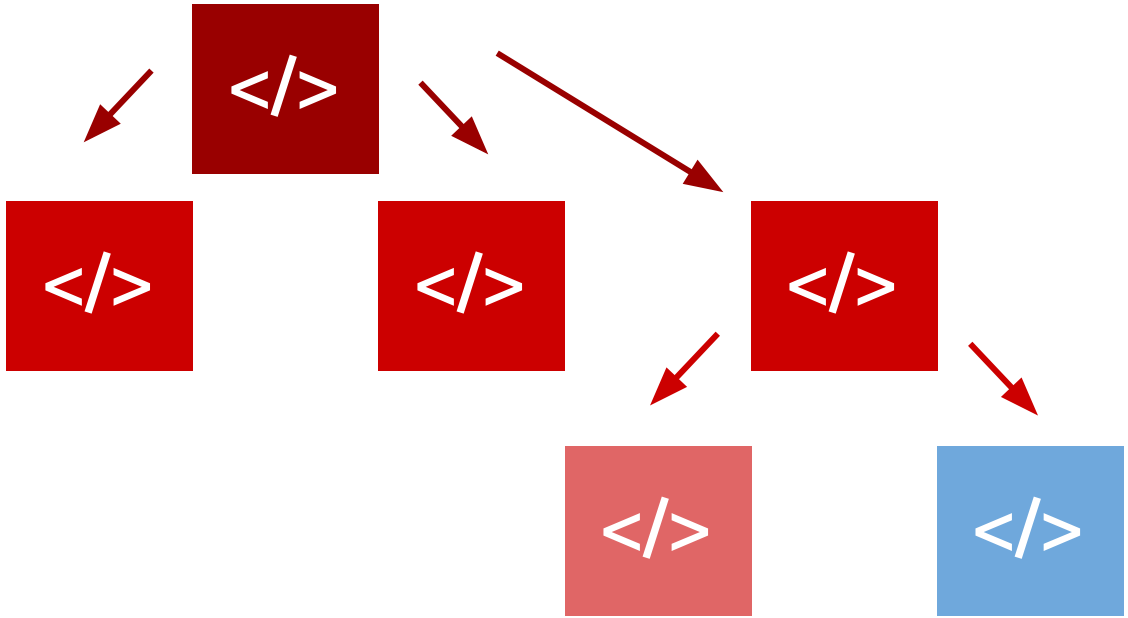
```
import { Component } from '@angular/core';
import * as saButton from '../webcomponents/sa-button/sa-button.js';

@Component({
  selector: 'sa-root',
  template: `
  <sa-button text="hi" ></sa-button>
`,
  styleUrls: ['./app.component.css']
})

export class AppComponent {...}
```

How do we **communicate** with the WC?

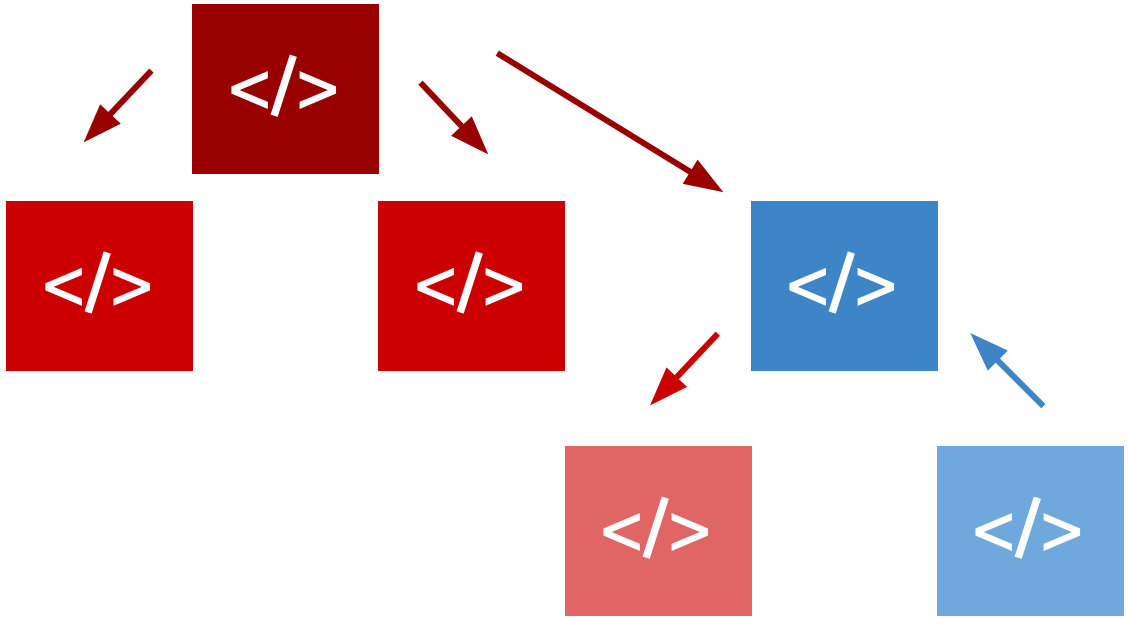
One way data flow



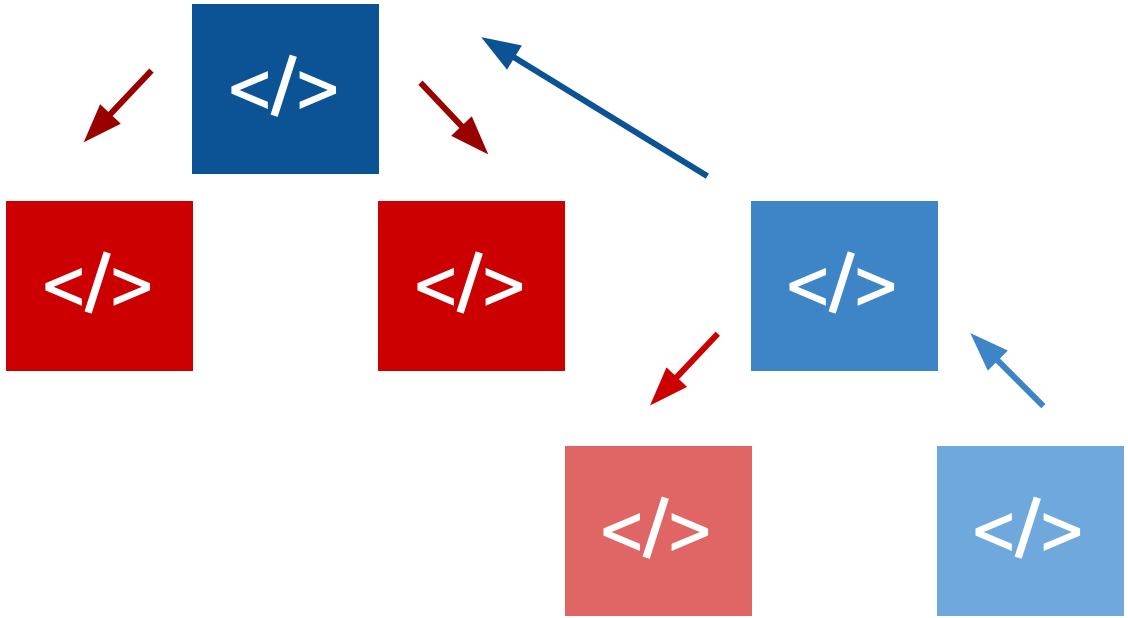
One way data flow



One way data flow



One way data flow



Sherry's Berries!!!

Strawberries

Fresh from our garden

ADD TO BASKET

Added: 2 Strawberries

REMOVE

Blackberries

Fresh from garden

ADD TO BASKET

Added: 0 Blackberries

REMOVE

Blueberries

Fresh from Sherry's garden!

ADD TO BASKET

Added: 0 Blueberries

REMOVE

```
<!doctype html>
<html lang="en">
 ▶<head>…</head>
 ▼<body>
  ▼<sa-root _nghost-c0 ng-version="7.0.4">
   ▼<sa-dashboard _ngcontent-c0 _nghost-c1>
    ▶<sa-header _ngcontent-c1 _nghost-c2>…</sa-header>
    ▼<div _ngcontent-c1 class="berries">
      <!--bindings={
        "ng-reflect-ng-for-of": "[object Object],[object
      Object"
      }-->
     ▼<sa-card _ngcontent-c1 _nghost-c3 ng-reflect-berry="
      [object Object]">
        <img _ngcontent-c3 class="shadow__image" id="shadow-
        image" src="https://bit.ly/2w8WQDF" alt=
        "Strawberries">
       ▶<div _ngcontent-c3 class="shadow__info" id="shadow-
        info">…</div>
       ▼<sa-counter _ngcontent-c3 _nghost-c4 ng-reflect-
        berry="[object Object]">
        ▼<div _ngcontent-c4 class="counter">
          ▼<sa-button _ngcontent-c4 style="--padding:5px; --
          font-size: 14px;" text="Add to basket" ng-version=
          "7.0.4"> == $0
            ▼#shadow-root (open)
              ▼<style>
                .btn{font-weight:var(--font-
                weight,200);font-size:var(--font-
                size,24px);background-color:var(--
                background-color,#cd5c5c);border:0;border-
                radius:5px;color:var(--
                color,#fff);padding:var(--
                padding,10px);text-transform:uppercase}
               </style>
               <button class="btn" id="text">Add to basket
               </button>
             </sa-button>
           ▶<p _ngcontent-c4>…</p>
           ▶<sa-button _ngcontent-c4 style="--padding:5px; --
```

… sa-root sa-dashboard div sa-card sa-counter div sa-button

Styles Event Listeners DOM Breakpoints Properties »

Filter :hov .cls +

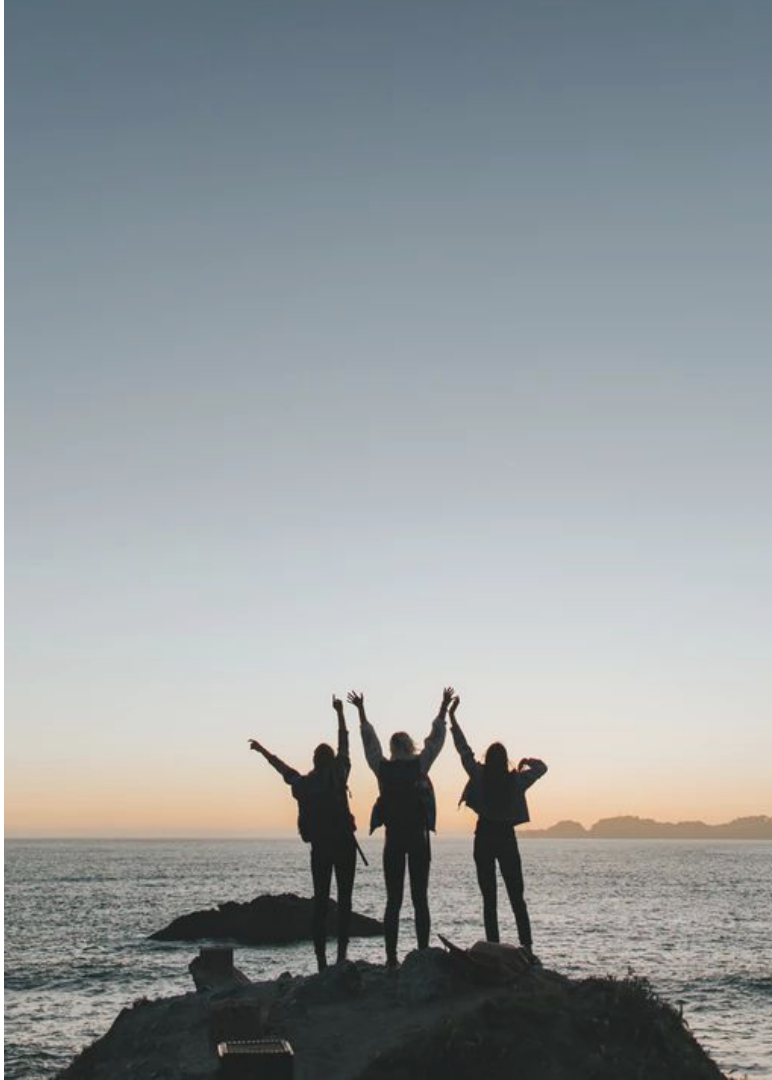
```
element.style {
  --padding: 5px;
```

margin -

border -

We are almost done!

- ✓ Create a web component with Angular
- ✓ Add a web component to an existing Angular project
- ✓ Use the web component in an Angular project




Do Web Components rock?
EXPLORE

Web Components DO Rock!

- Maximum interoperability
- Support from Frameworks (Tooling)
- Many success stories
- Browsers are ready

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

*https://webcomponents.org

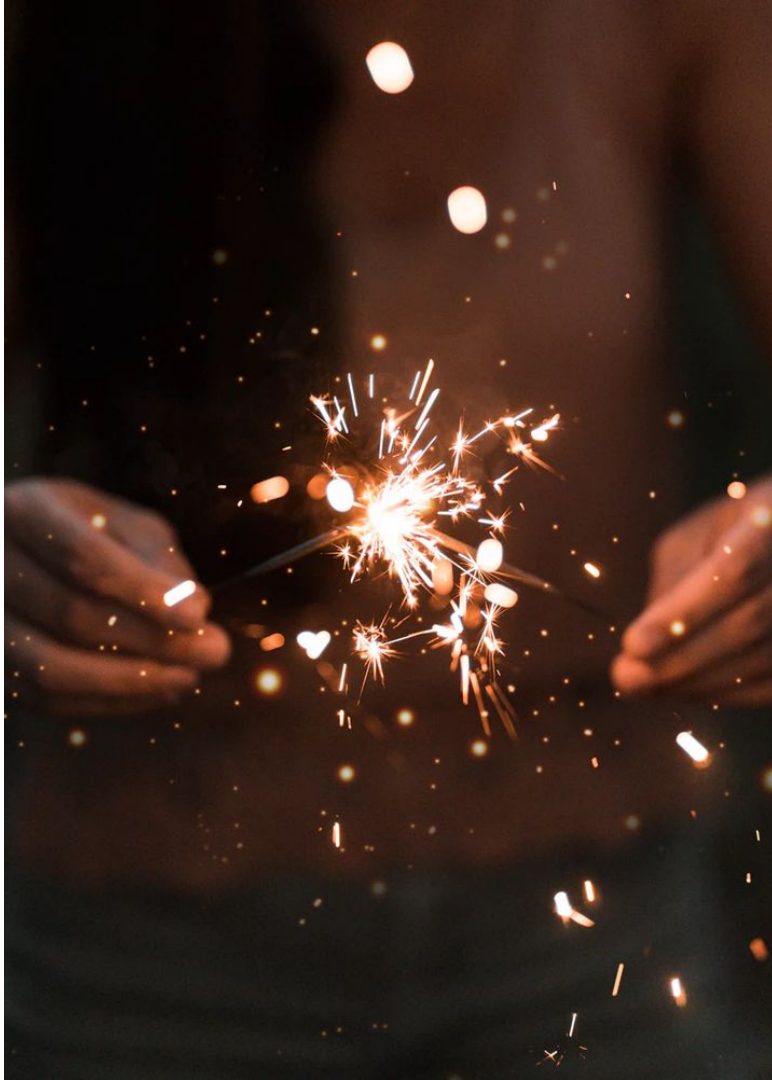
What’s next?

lit-html

Check out: https://youtu.be/lo6JjgckHbg

What is lit-html?

- <template> in JS with template literals
- Extremely fast (Partial update)
- Customizable and extensible (Directives)



Simple button

Here is a simple web component button

N/A

Elements Console Sources Network Performance Memory

```
<!doctype html>
<html class=" ">
 ▶ <head>…</head>
 ▼ <body>
   ▼ <div class="main__container">
       <h1>Simple button</h1>
       <p>Here is a simple web component button</p>
     ▼ <sa-button text style="--background-color: navy; --color: aqua; --font-weight: 600; --
     padding: 12px;" onclick="saFunction()">
       ▼ #shadow-root (open)
         ▶ <style>…</style>
           <button class="btn" id="n/a">n/a</button> == $0
       </sa-button>
     </div>
   </body>
 </html>
```

html body div.main__container sa-button #shadow-root button#n/a.btn

Styles Event Listeners DOM Breakpoints Properties Accessibility $scope ADT Properties

Filter :hov .cls +

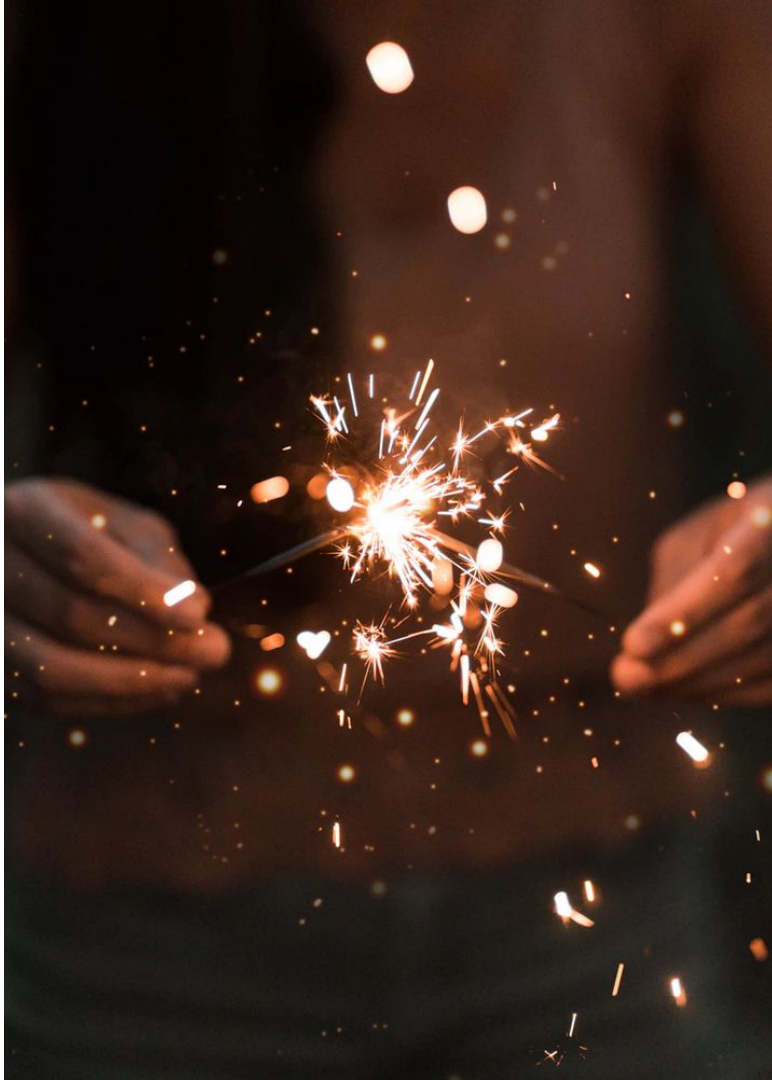
element.style {

lit-element

Check out: https://youtu.be/ypPRdtjGooc

What is lit-element?

- Base class for creating web components
- Uses lit-html to render into the element's Shadow DOM
- React to changes
- Extremely fast & light



Simple button

Here is a simple web component button

CLICK

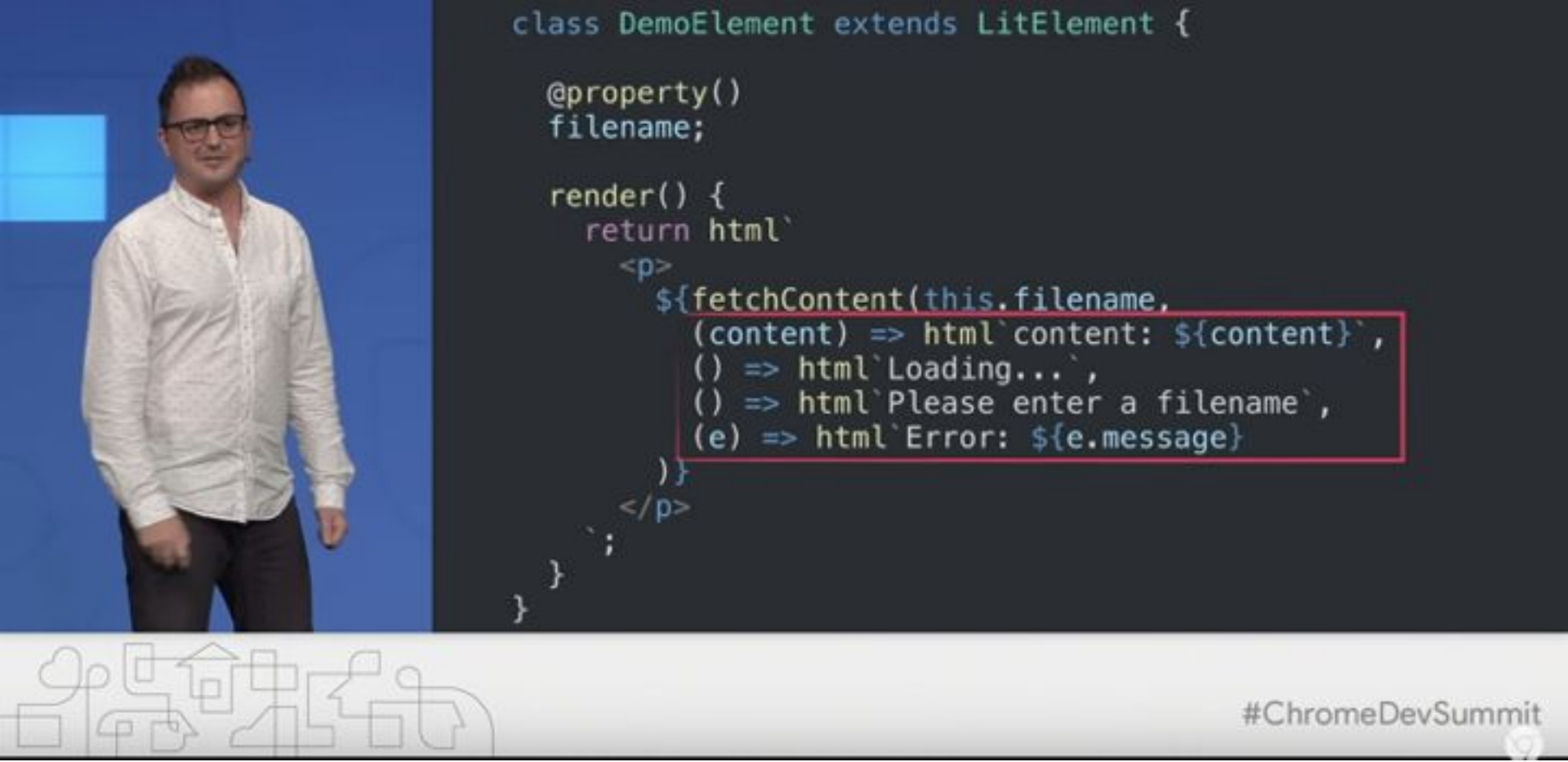

```
<!doctype html>
<html class=" ">
▶<head>…</head>
▼<body> == $0
  ▼<div class="main__container">
    <h1>Simple button</h1>
    <p>Here is a simple web component button</p>
    ▼<sa-button title="Click" style="--background-color: navy; --color: aqua; --
    font-weight: 600; --padding: 12px;" onclick="saFunction()">
      ▼#shadow-root (open)
        <!---->
        ▶<style>…</style>
        ▼<button id="text" class="btn">
          <!---->
          "Click"
          <!---->
        </button>
        <!---->
    </sa-button>
  </div>
</body>
</html>
```

Elements Console Sources Network Performance

html body

Styles Event Listeners DOM Breakpoints Properties Accessibility $scope

Filter :hov .cls +



Check out: https://youtu.be/ypPRdtjGooc

Theming

CSS Shadow Parts

::part()

Check out: https://meowni.ca/posts/part-theme-explainer/

::part

```
<x-foo>
 #shadow-root
  <div part="some-box"><span>...</span></div>
  <input part="some-input">
  <div>...</div> /* not styleable
</x-foo>
```

Source: https://meowni.ca/posts/part-theme-explainer/

::part

At a document which has <x-foo>:

```
x-foo::part(some-box) { … }
x-foo::part(some-box):hover { … }
x-foo::part(some-input)::placeholder { … }
```

Source: https://meowni.ca/posts/part-theme-explainer/

Angular ♥ Web Components

http://bit.ly/front-end-love-web-components

Thank you!

Sherry List
@SherrryLst

Ana Cidre
@AnaCidre_

Best practices

- https://w3ctag.github.io/webcomponents-design-guidelines/
- https://github.com/webcomponents/gold-standard/wiki
- https://developers.google.com/web/fundamentals/web-components/best-practices

Resources

- https://www.softwarearchitekt.at/post/2018/07/13/angular-elements-part-i-a-dynamic-dashboard-in-four-steps-with-web-components.aspx
- https://meowni.ca/posts/part-theme-explainer/
- https://medium.com/google-developer-experts/are-web-components-a-thing-5a116b1da7e4
- https://www.telerik.com/blogs/web-components-101-an-introduction-to-web-components
- https://www.softwarearchitekt.at/post/2019/01/27/building-angular-elements-with-the-cli.aspx