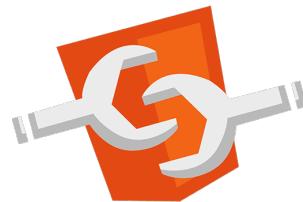


Demystifying Web Components

Perfect Weapon for the Convergence of Web





ng-India



Rahat Khanna
Twitter: @mappmechanic

Currently @ Apple
Ex- Flipkart, Genpact

#ngIndia

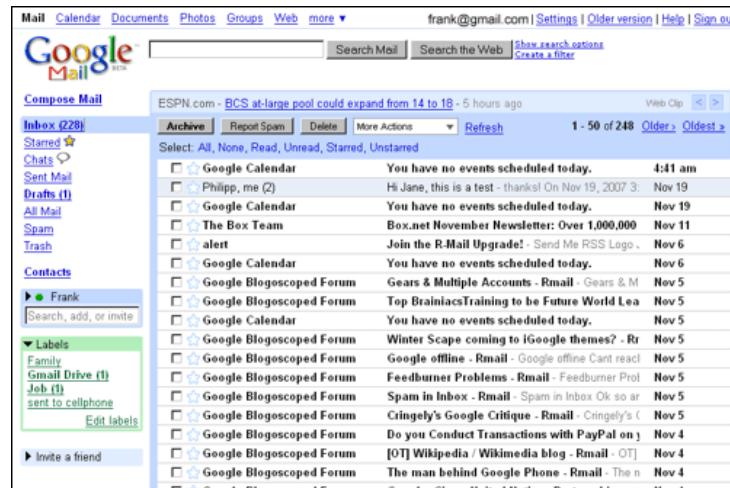
www.ng-ind.com

Feb 24, 2018

Agenda

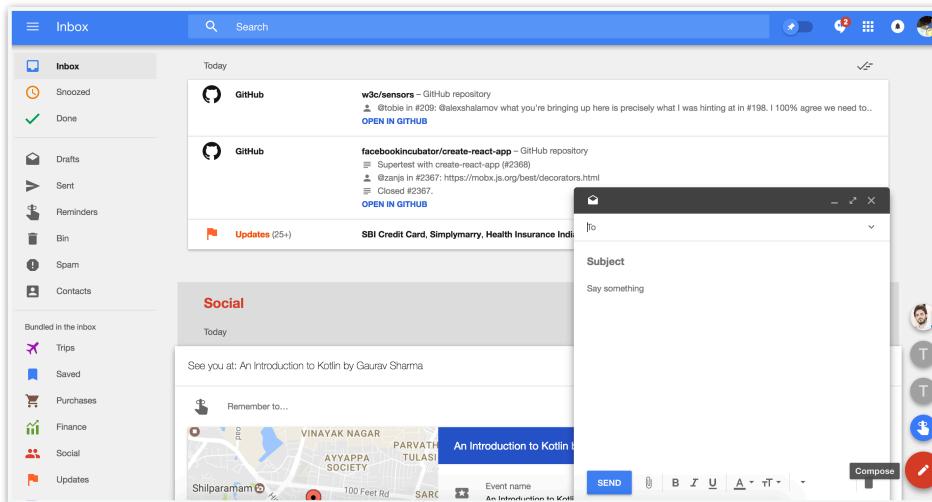
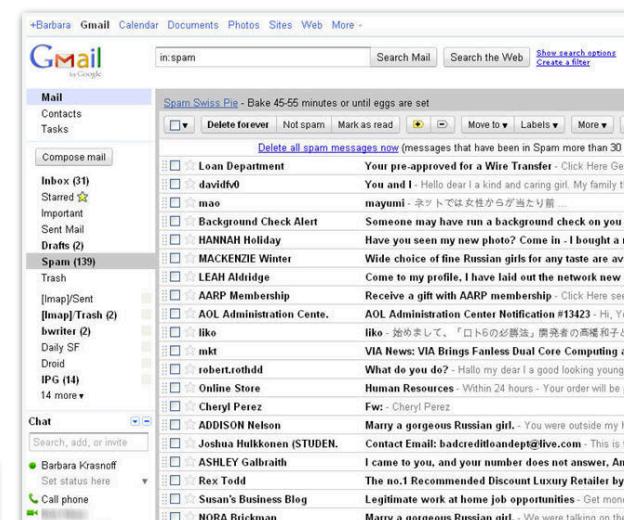
- Problems in the current Front End World
- Getting Started with Web Components
- 4 Pillars of Web Components
- Using a natively developed Web Component with an Angular App

Evolution of Front End Dev

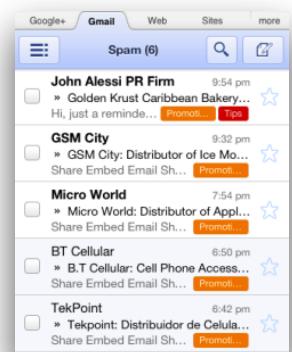


Static HTML + CSS
Served from Server

Ajax, DHTML, CSS
Mobile Web, Responsive



SPA, UI/UX, CSS3 Animations
PWAs, Server Rendering,
Bundling, CDN



Problems



Problems

1. Learn a new framework every 6 months

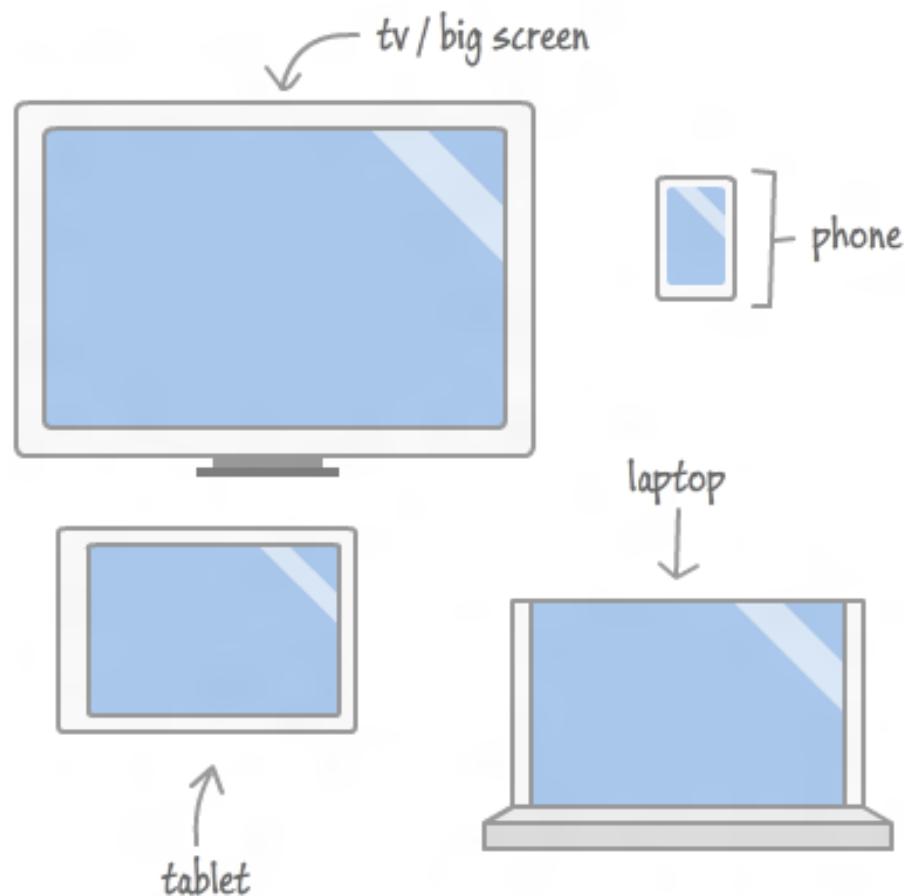
Front-End Framework Extravaganza!

A highly opinionated, ghastly incomplete, category-mistake-littered, but *fun* overview of available Front-End Frameworks (and tools). | @jeroenheijmans | <http://fefe.jeroenheijmans.nl> | v1.0.0

Lodash	Less	Sass	Angular	Knockout	React	Rx	Bacon	SignalR
Underscore	JSON2	moment	Backbone	mustachejs	Ractive	Sammy	Socket.io	Breeze
Modemizr	Require	Browsify	Ember	Mithril	Aurelia	Google Maps	OpenLayers	jVectorMap
HTML5 Boilerplate	Webpack	Systemjs	Dojo	Prototype	MooTools	Google Fonts	Typekit	Cufon
HTML5 Doctor CSS reset	QUnit	Jasmine	YUI	Ext	jQuery	Reveal	Impress	Deck
Normalize.css	Mocha	Sinon		Kendo	jQuery UI	Colorbox	Fancybox	Lightbox
Eric Meyer Reset.css	Select2	Chosen	Hammer	Sencha	jQuery Mobile	Highcharts	Flot	FusionCharts
Pure	Selectize		Ionic	Cordova		NVD3	Chartist	Chart.js
Polymer	cssdesk.com	plnkr.co	Glyphicon	Font Awesome	Elusive	D3	Snap	Three
Foundation	cssdeck.com	jsfiddle.com				Raphael	svgjs	Pixi
Unsemantic	dabblet.com	codepen.io	NicEdit	MarkItUp		Fabric	Paper	Babylon
Semantic UI	bootply.com	jsbin.com	CKeditor	TinyMCE	AlohaEditor	Processing	Scene	Famous

Problems

2. Device sizes and Browser quirks

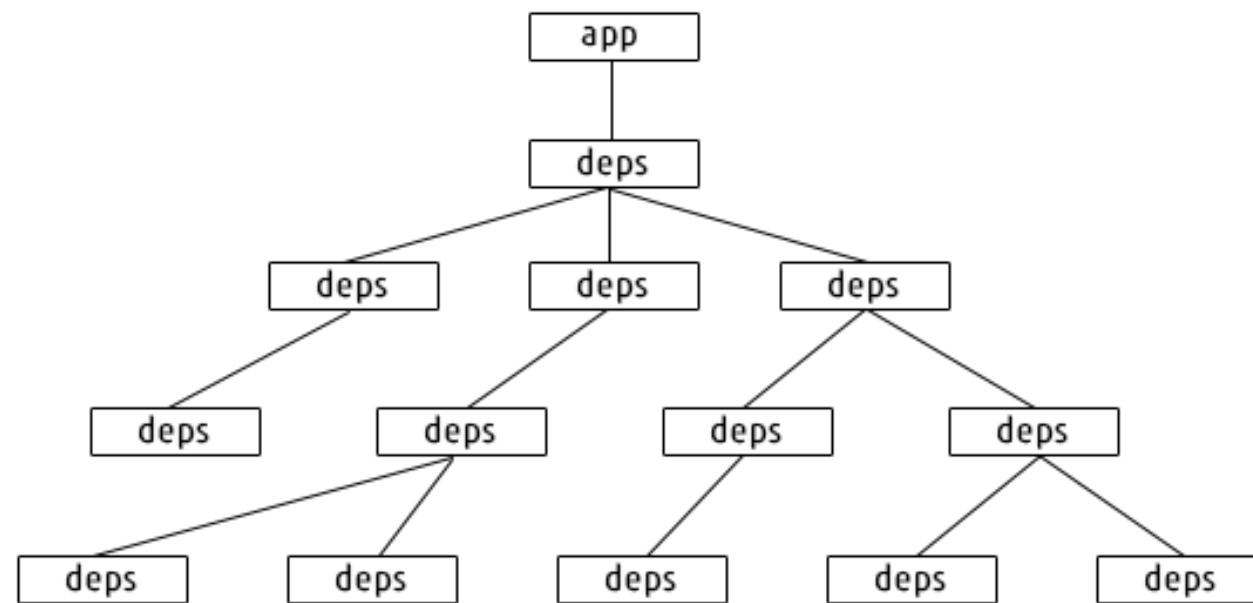


Problems

3. Dependency management - node js dependency hell

nested dependencies

npm (node_modules)

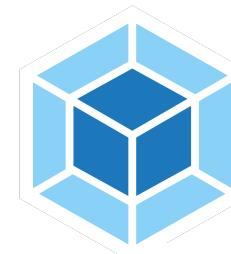
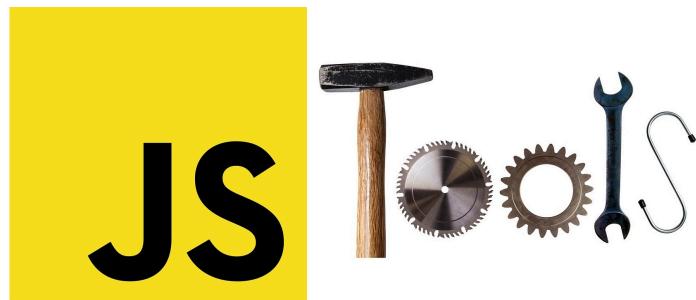


Problems

4. Build Management & deployment to CDNs

```
Version: webpack 1.12.9
Time: 31ms
    Asset      Size  Chunks      Chunk Names
index.html  165 bytes          [emitted]

ERROR in ./app/app.js
Module parse failed: c:\MMDATA\Gonzalo-Dev\gonzalobarba.com\app\app.js Line 1: Unexpected token
You may need an appropriate loader to handle this file type.
| import React, {Component} from 'react';
| import {render} from 'react-dom';
|
webpack: bundle is now VALID.
```



webpack

BABEL



or



Problems

5. Referencing of HTML5, CSS & JS separately for reusable components

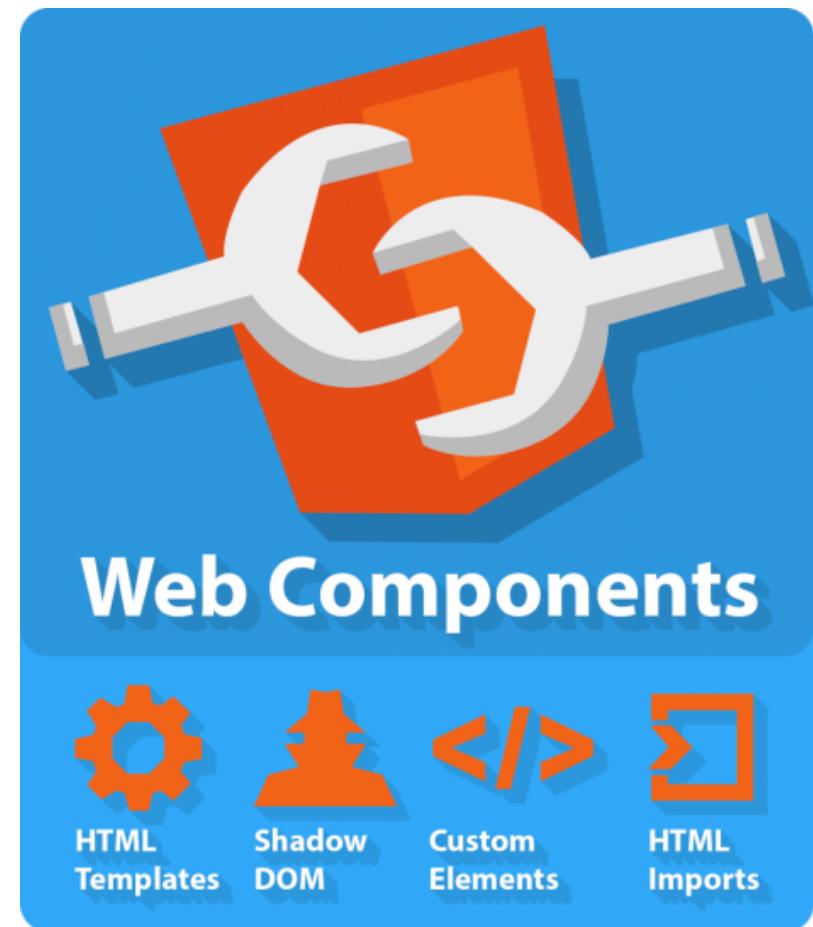
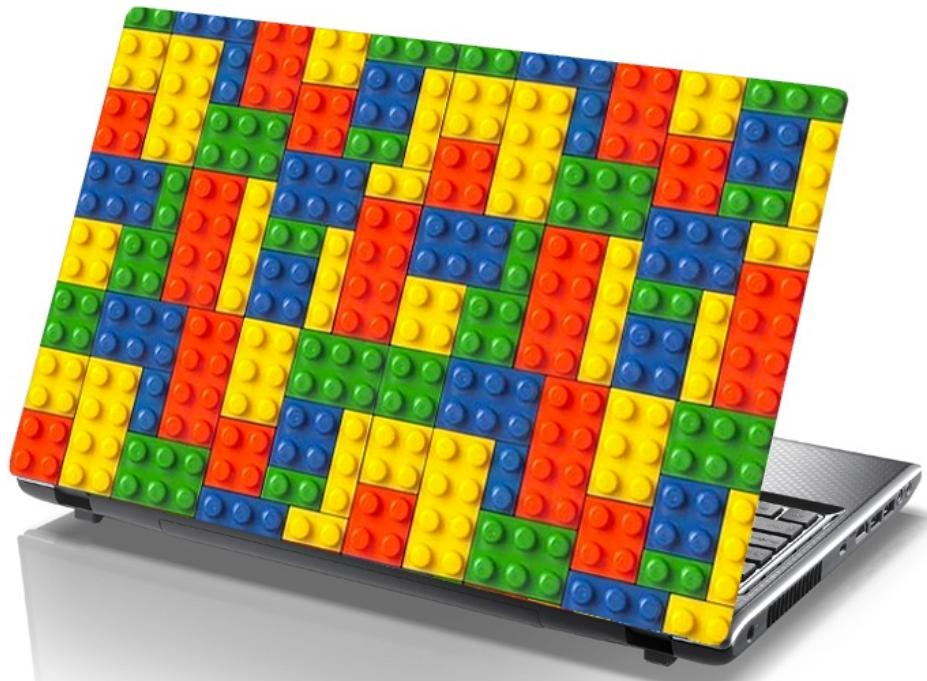
```
4  <head>
5      <title>Angularjs Component</title>
6      <meta charset="UTF-8">
7      <script type="text/javascript" src="lib/jquery/jquery-2.1.1.min.js"></script>
8      <script type="text/javascript" src="lib/bootstrap/js/bootstrap.min.js"></script>
9      <script type="text/javascript" src="lib/angularjs/angular.min.js"></script>
10     <script type="text/javascript" src="lib/angularjs/angular-resource.min.js"></script>
11     <script type="text/javascript" src="lib/angularjs/angular-animate.min.js"></script>
12     <script type="text/javascript" src="lib/angularui/ui-bootstrap-tpls-0.11.0.min.js"></script>
13     <script type="text/javascript" src="lib/angularui/angular-ui-router.min.js"></script>
14     <script type="text/javascript" src="lib/angularui/ng-grid/ng-grid-2.0.11.min.js"></script>
15     <script type="text/javascript" src="lib/underscore/underscore-min.js"></script>
16     <script type="text/javascript" src="lib/utils/spin.min.js"></script>
17     <script type="text/javascript" src="lib/utils/jquery.bpopup.min.js"></script>
18
19     <script type="text/javascript" src="js/app.js"></script>
20     <script type="text/javascript" src="js/controllers.js"></script>
21     <script type="text/javascript" src="js/directives.js"></script>
22     <script type="text/javascript" src="js/filters.js"></script>
23     <script type="text/javascript" src="js/services.js"></script>
24     <script type="text/javascript" src="js/utils.js"></script>
25     <script type="text/javascript" src="config/config.js"></script>
26     <script type="text/javascript" src="test/restful/restful.js"></script>
27
28     <!-- tabs controller -->
29     <script type="text/javascript" src="js/navs/navs.js"></script>
30     <script type="text/javascript" src="js/navs/nav1.js"></script>
31     <script type="text/javascript" src="js/navs/nav2.js"></script>
32     <script type="text/javascript" src="js/navs/nav3.js"></script>
33
34     <link rel="stylesheet" href="lib/bootstrap/css/bootstrap.min.css">
35     <link rel="stylesheet" href="lib/angularui/ng-grid/ng-grid.min.css">
36     <link rel="stylesheet" href="css/app.css">
37 </head>
38
```

Problems are not **STOP** signs,
they are **guidelines**

You already know Web Components ? Don't you ?



Web Components as Weapon for Web Convergence



History of Web Components

2013 - first mentioning of WC by Google - v0

2014 - added official support v0 in Chrome & Opera



2015 - several meetings to discuss what goes to v1
and what to further versions

2016 - added official support of custom elements v1
in Chrome and Opera

2017 - Shadow DOM support added in Safari
(desktop & mobile) and is coming to Firefox soon.

Architecture



JS Code for Component Behaviour

ES6 Classes (mostly)

Web Components Code

HTML



CSS

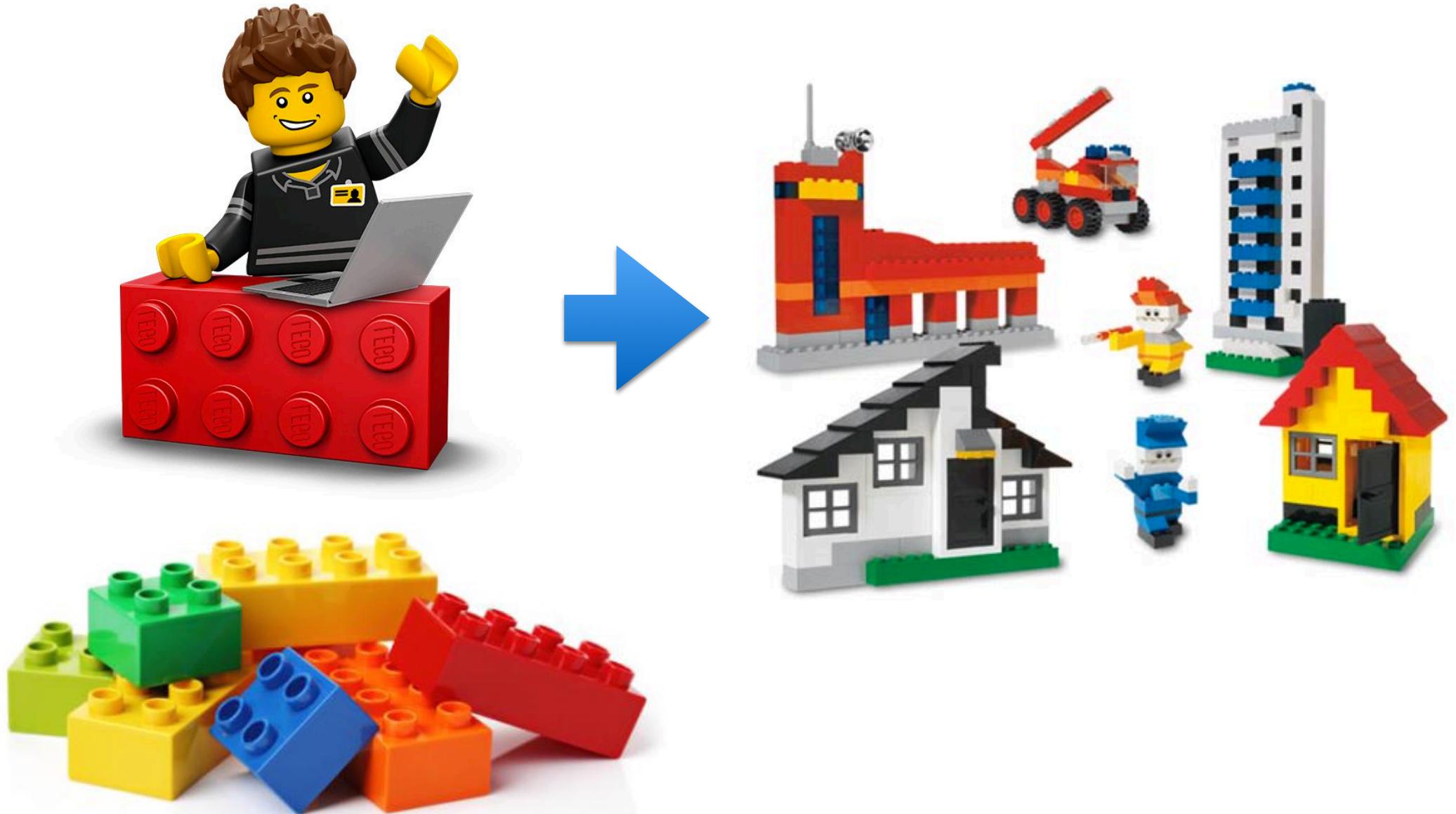


HTML Templates + CSS Styles

Polyfills like PolymerJS, X-Tag
which are temporary

Browser APIs - like Custom Elements v1, Shadow DOM, HTML Imports

Why & Where should we use Web Components ?



Custom Elements

Create new HTML tags or **Extend** existing ones
Foundation of **web components**

```
<my-app>
  <my-header>
    <h1>App Title</h1>
  </my-header>
  <my-content>
    <form>
      Enter name<input type="text">
      <my-button>Submit</my-button>
    </form>
  </my-content>
</my-app>
```

Creating a Simple Custom Element

```
// ES2015 Class for the new Custom Element
class MyButton extends HTMLElement {
    constructor() {
        console.log('my button has been initialised.');
    }

    // Getter & Setter for Properties
    get color() {
        return this._color;
    }

    set color(value) {
        // Reflect the value of the open property as an HTML attribute.
        if (val) { this.setAttribute('color', value); } else { this.removeAttribute('color'); }
        this.updateColor();
    }

    updateColor() {
    }
}

customElements.define('my-button', MyButton);
```

Lifecycle Methods - Callbacks

constructor()

Called when the element is created or upgraded

connectedCallback()

Called when the element is inserted into a document, including into a shadow tree

disconnectedCallback()

Called when the element is removed from a document

attributeChangedCallback(attributeName, oldValue, newValue, namespace)

Called when an attribute is changed, appended, removed, or replaced on the element.

Only called for [observed attributes](#).

adoptedCallback(oldDocument, newDocument)

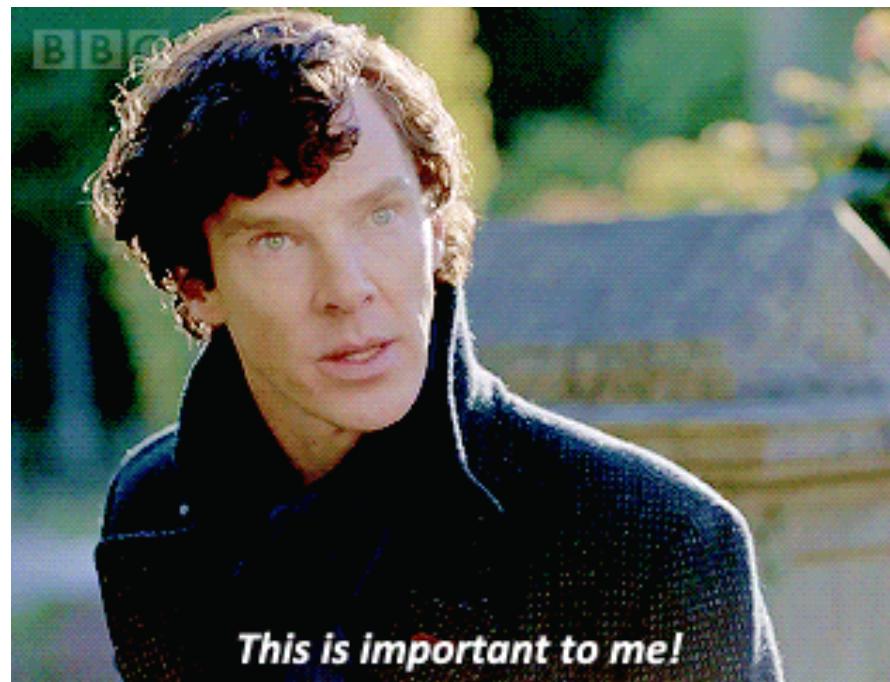
Called when the element is adopted into a new document



Shadow DOM

Shadow DOM is a new specification which will help developers to keep **some HTML & CSS code separate from the rest of the page.**

color: #fff !important;



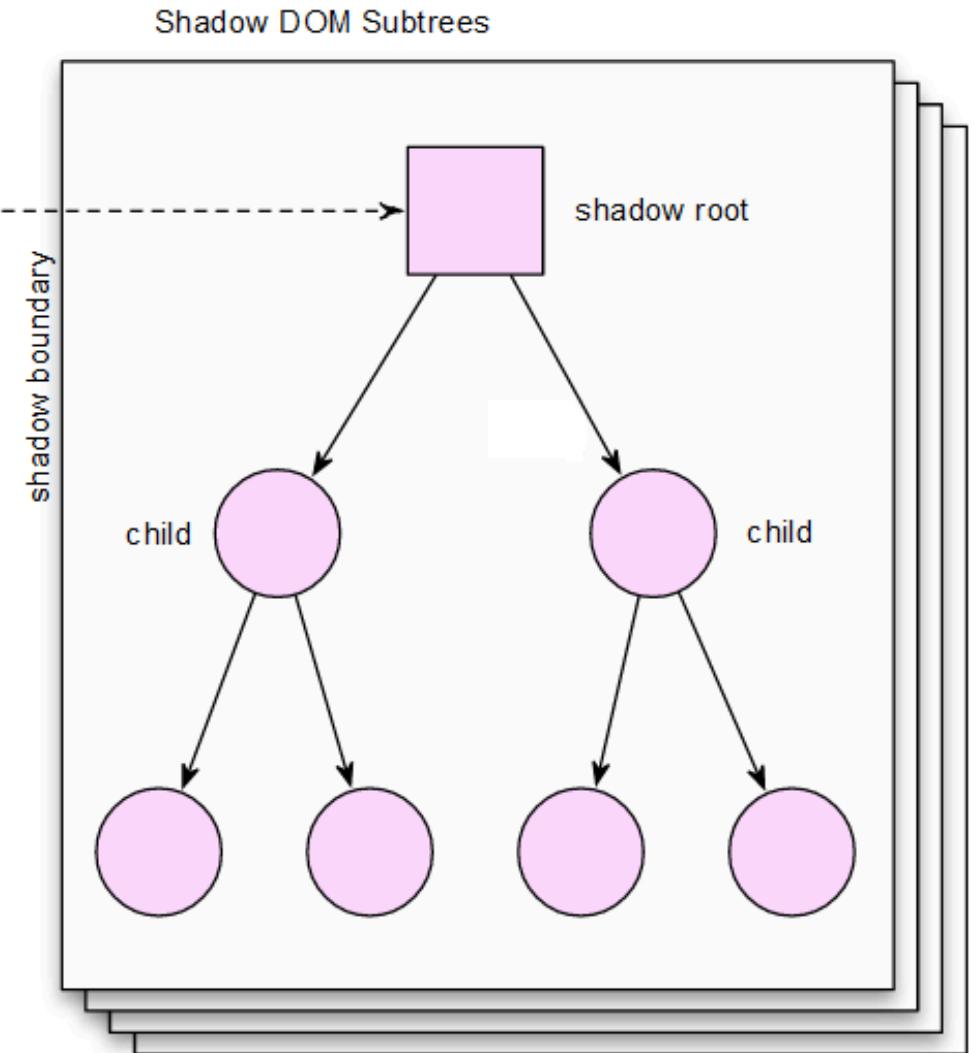
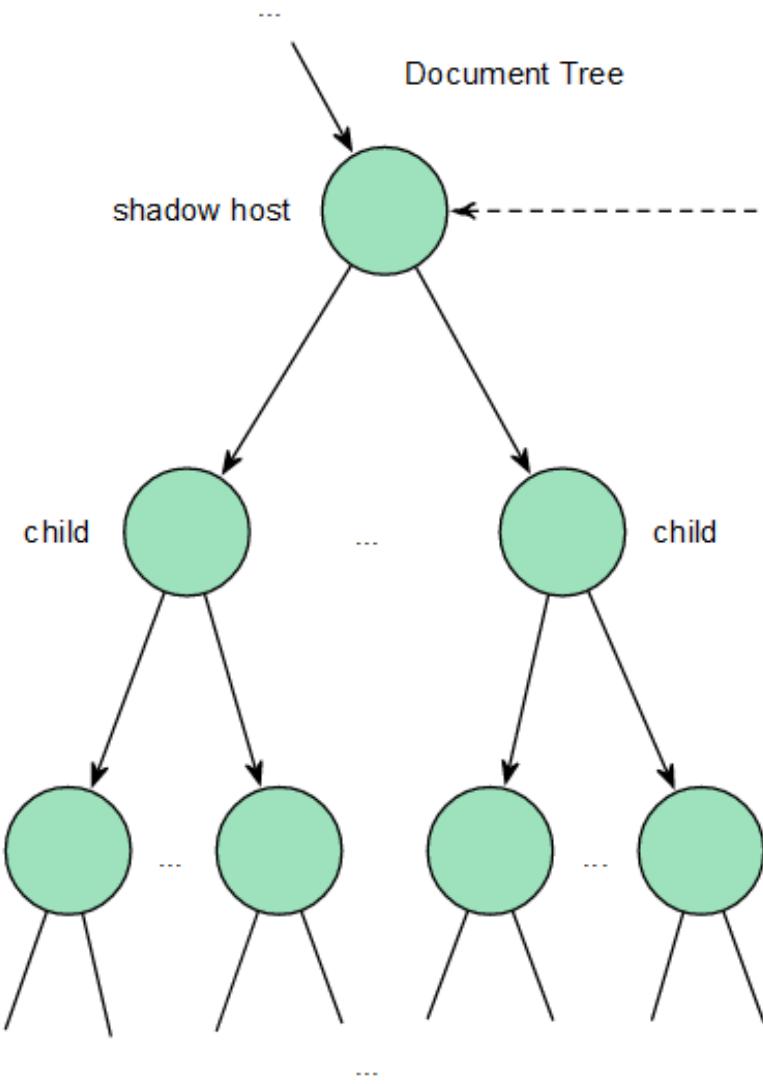
Benefits of Shadow DOM

- **Isolated DOM** - self contained component DOM
- **Scoped CSS** - css defined inside shadow DOM
- **Composition** - declarative markup with hidden DOM
- **Simplifies CSS** - simple css selectors without conflicts
- **Productivity** - build using reusable components

Shadow DOM (contd)

- Shadow DOM is attached to an existing element
 - Either a native element like <div> or <p>
 - Or a Custom Element like <my-element>
- Style Shadow DOM like normal CSS and few extra selectors
 - Pseudo-classes- :host, :host-context()
 - Pseudo-elements- ::slotted()
 - Combinator- >>>*

Shadow DOM Diagram



HTML Template Element

The **HTML <template> element** is a mechanism for holding client-side content that is not to be rendered when a page is loaded but may subsequently be instantiated during runtime using JavaScript.

HTML Template Element

Example

```
1 <table id="producttable">
2   <thead>
3     <tr>
4       <td>UPC_Code</td>
5       <td>Product_Name</td>
6     </tr>
7   </thead>
8   <tbody>
9     <!-- existing data could optionally be included here -->
10  </tbody>
11 </table>
12
13 <template id="productrow">
14   <tr>
15     <td class="record"></td>
16     <td></td>
17   </tr>
18 </template>
```

Slot Element - Light DOM

```
<slot> </slot>
```

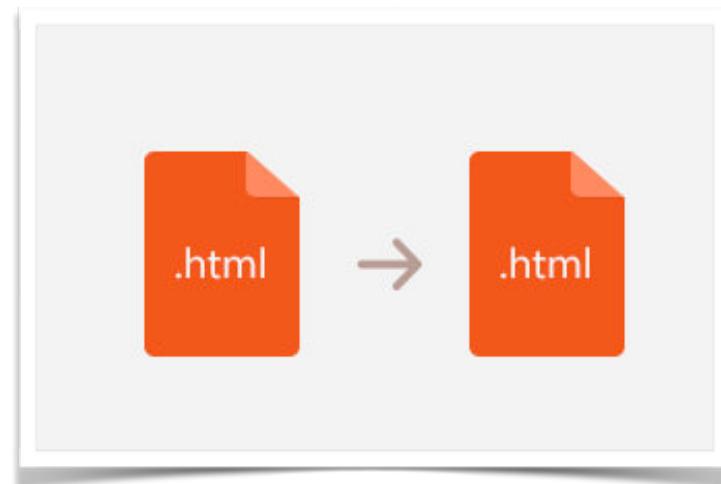
```
<slot name='icon'> </slot>
```

<slot> is a placeholder where child nodes will render

HTML Imports (not recommended)

HTML Imports are a way to include and reuse HTML documents in other HTML documents.

```
<link href="extern.html" rel="import" />
```

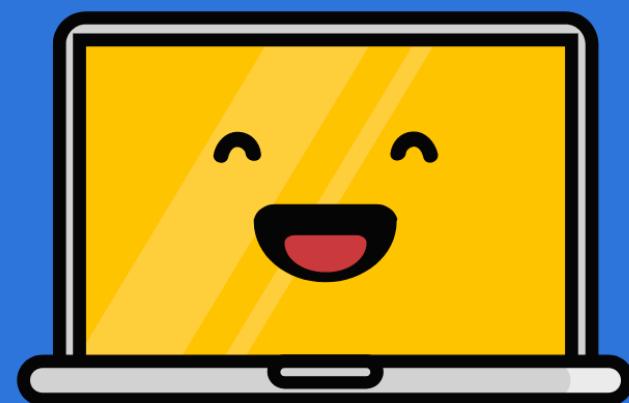


Browser Compatibility

Browser support	CHROME	OPERA	SAFARI	FIREFOX	EDGE
 TEMPLATES	 STABLE	 STABLE	 STABLE	 STABLE	 STABLE
 CUSTOM ELEMENTS	 STABLE	 STABLE	 STABLE	 POLYFILL  DEVELOPING	 POLYFILL  CONSIDERING
 SHADOW DOM	 STABLE	 STABLE	 STABLE	 POLYFILL  DEVELOPING	 POLYFILL  CONSIDERING
 <SCRIPT TYPE="MODULE">	 STABLE	 STABLE	 10.1	 FLAG IN 54	 FLAG IN 15
 HTML IMPORTS	 STABLE	 STABLE	 POLYFILL  ON HOLD	 POLYFILL  ON HOLD	 POLYFILL  CONSIDERING

Source: webcomponents.org

Custom Elements Everywhere !



Angular

SCORE
30/30

CanJS

SCORE
26/30

Preact

SCORE
24/30

ReactJS

SCORE
16/30

Vue

SCORE
30/30

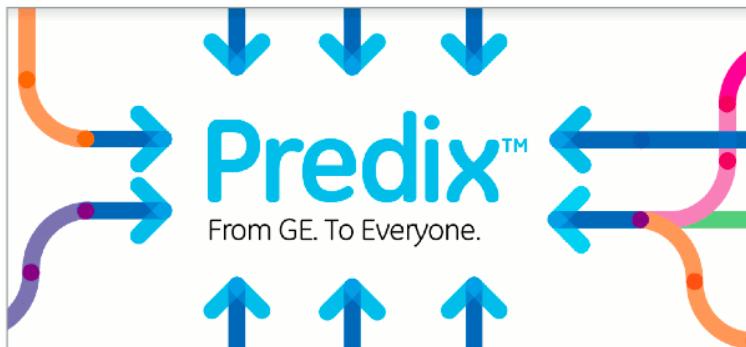
Courtesy: Rob Dodson Session

[Polymer Summit 2017](#)

WC with Angular Demo



Already using Web Components in Production



vaadin }>



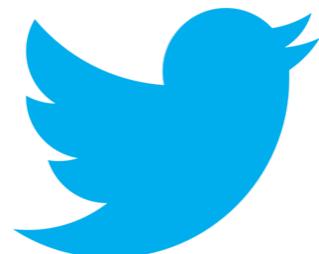
Bloomberg



Using Web Components with Angular



<http://bit.ly/WCSuperButton>



@mappmechanic

Thank You

E: yehtechnologies@gmail.com