But there is no web component for that!

Horacio Gonzalez
@LostInBrittany

Who am I ?

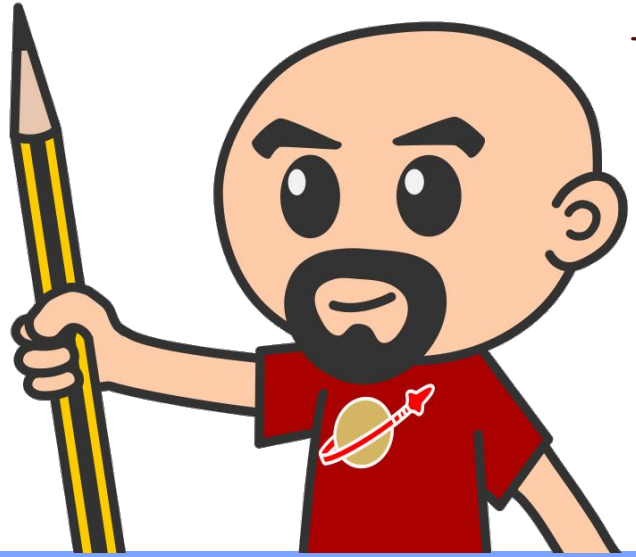
Horacio Gonzalez

@LostInBrittany

Cityzen Data

http://cityzendata.com

Spaniard lost in Brittany, developer, dreamer and all-around geek

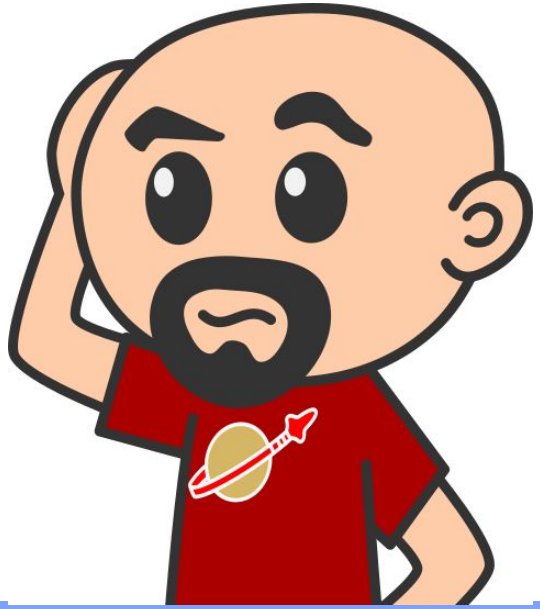


There is no webcomponent for that!

So there is no web component
for your nifty feature…

But there is a JS library

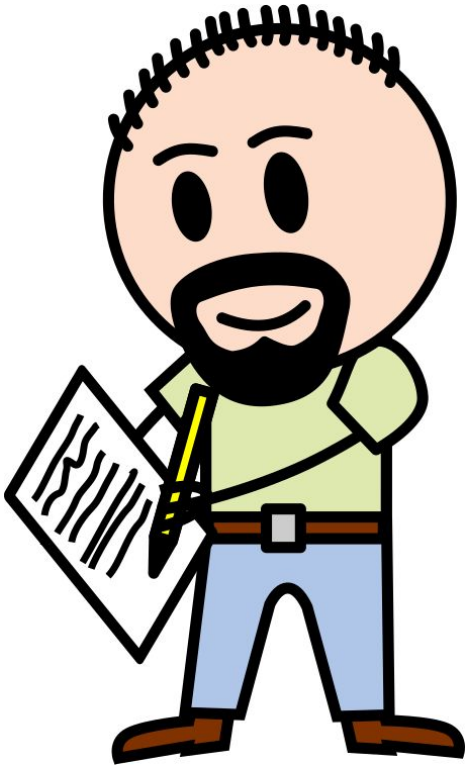
What can I do?



The show must go on!

Not having a component
for a feature
isn't a show stopper.

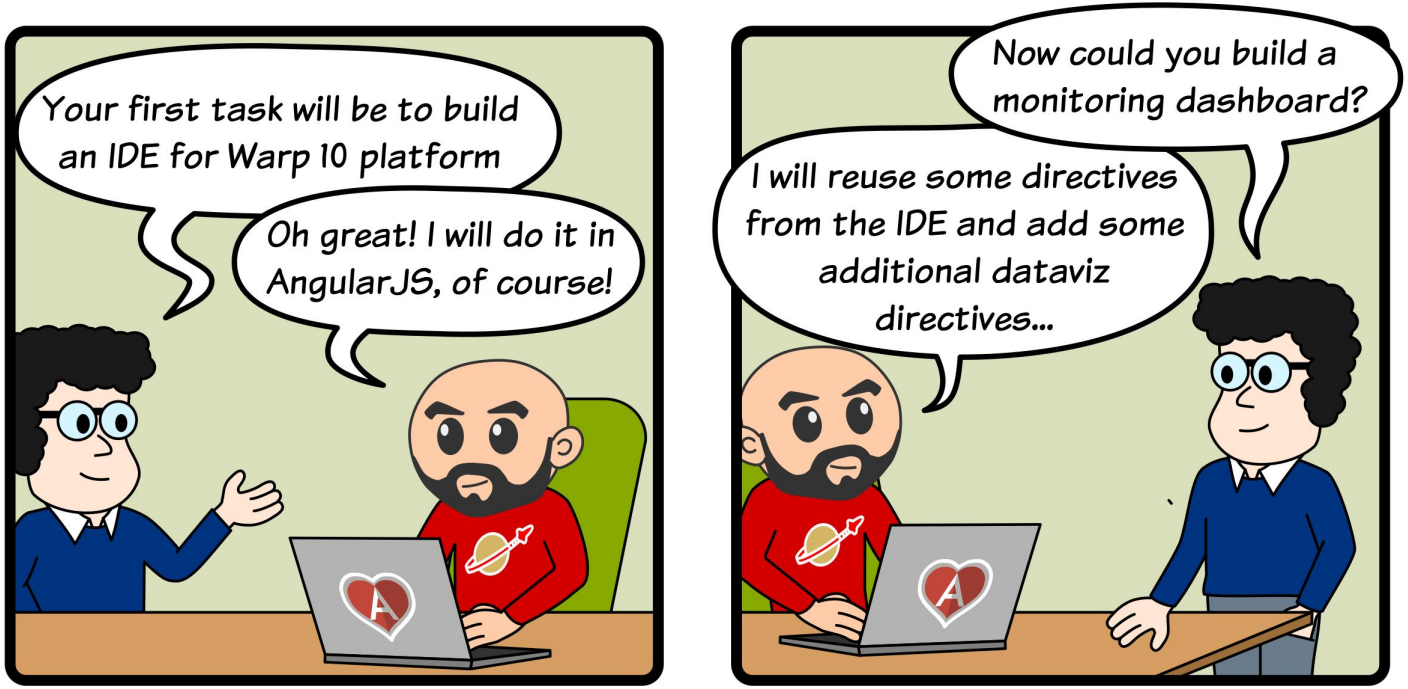
Writing it is way simpler
than you could think



Introduction

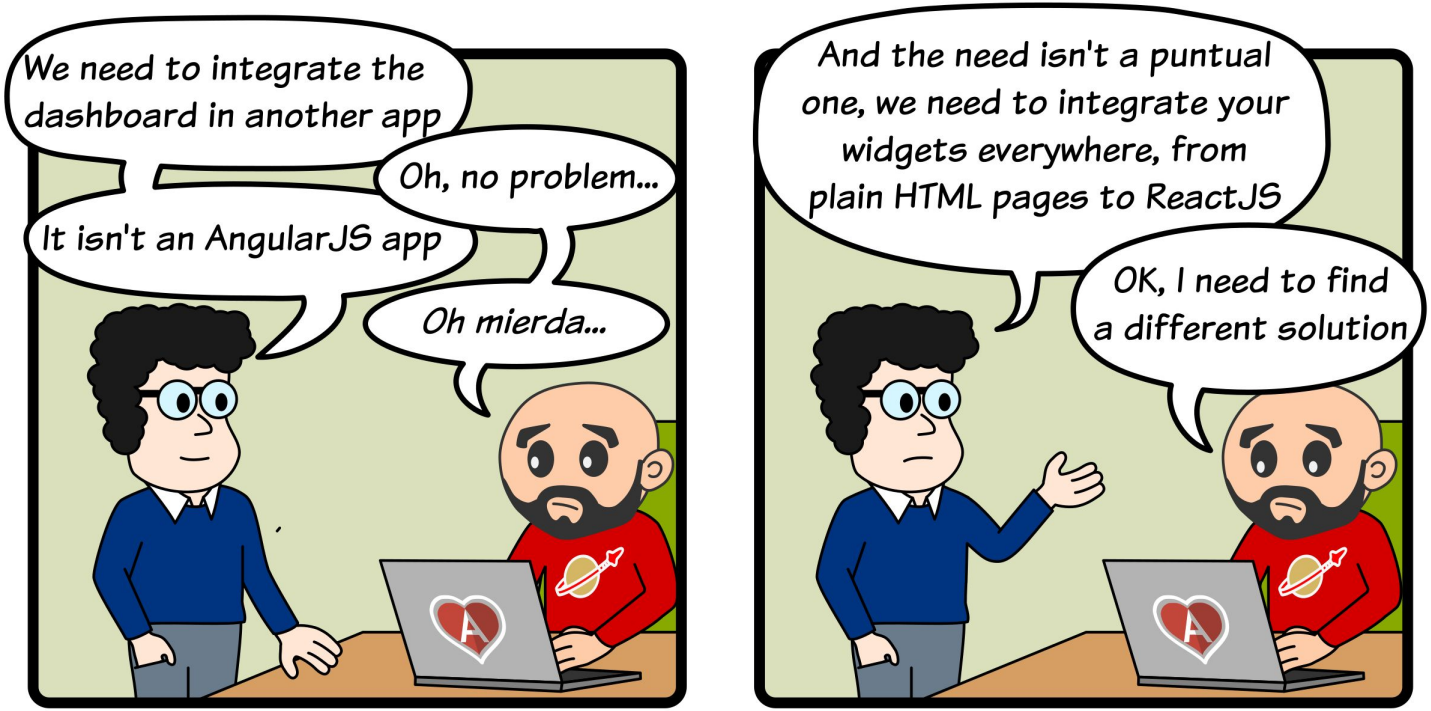
Context is everything

I was kinda an AngularJS fanboy



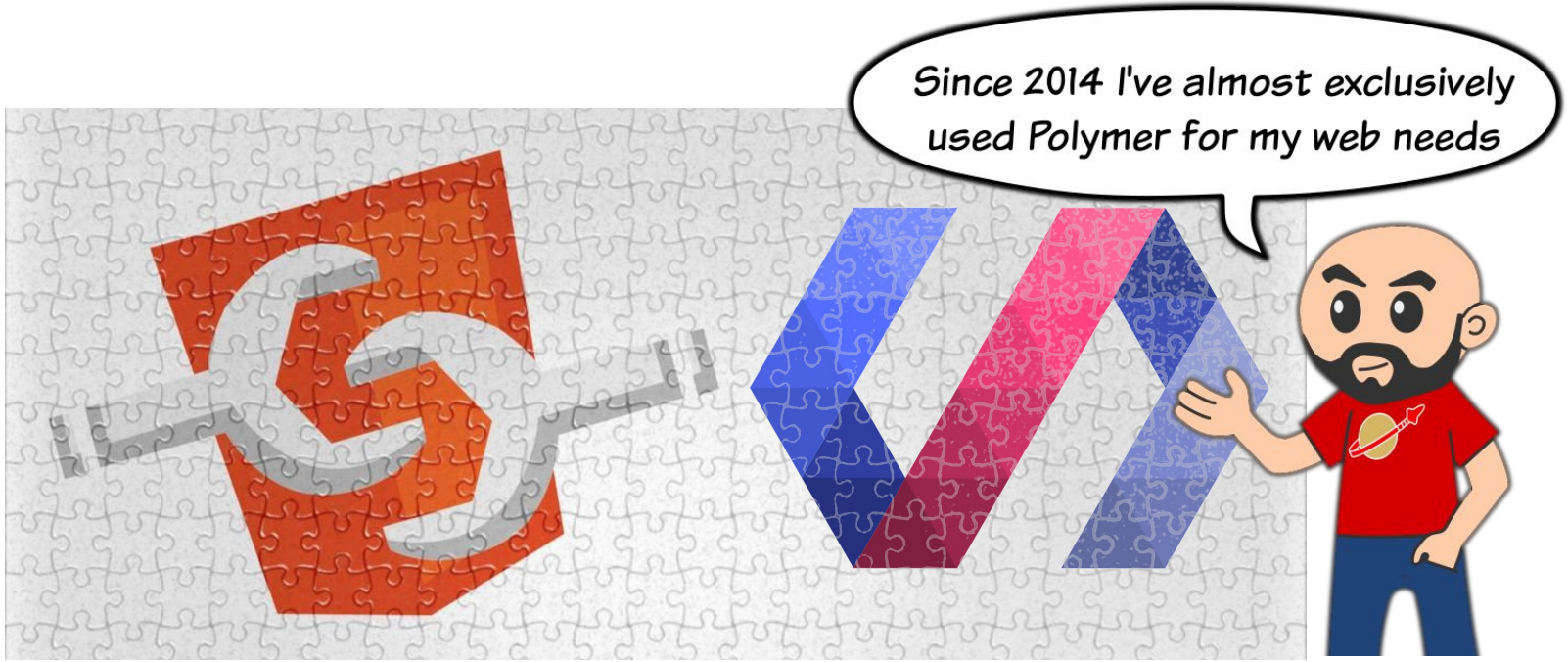
WARP 10

Until I hit a wall



WARP 10

Enter Web Components & Polymer



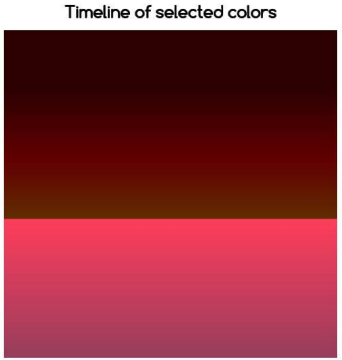
WebComponents, a modular approach to webapps

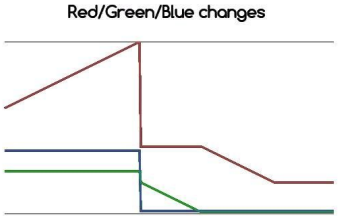
Are you sure you want to do it?

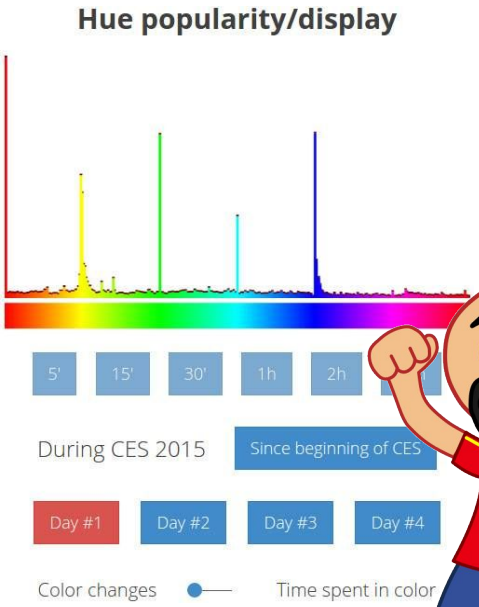
Don't do it, crazy Spaniard, it isn't production ready!

And it worked!

WARP 10







We put our first Polymer app in production on 2014 with Polymer 0.4

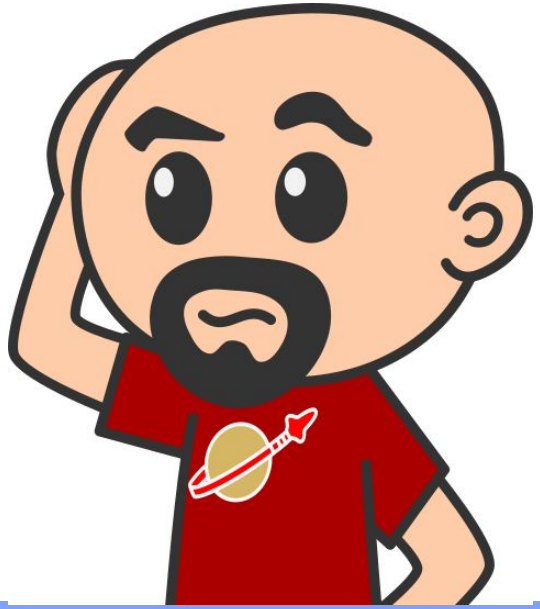
Full story: http://blog.cityzendata.com/2015/02/07/behind-CES-colors/

It was there I met the problem...

I used D3.js, NVD3 and canvas for my dataviz

But there was nothing like that in Polymer

What could I do?

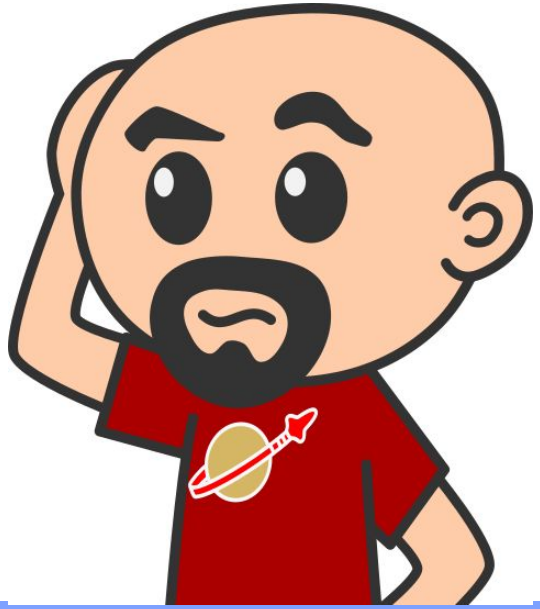


For each problem there is a solution

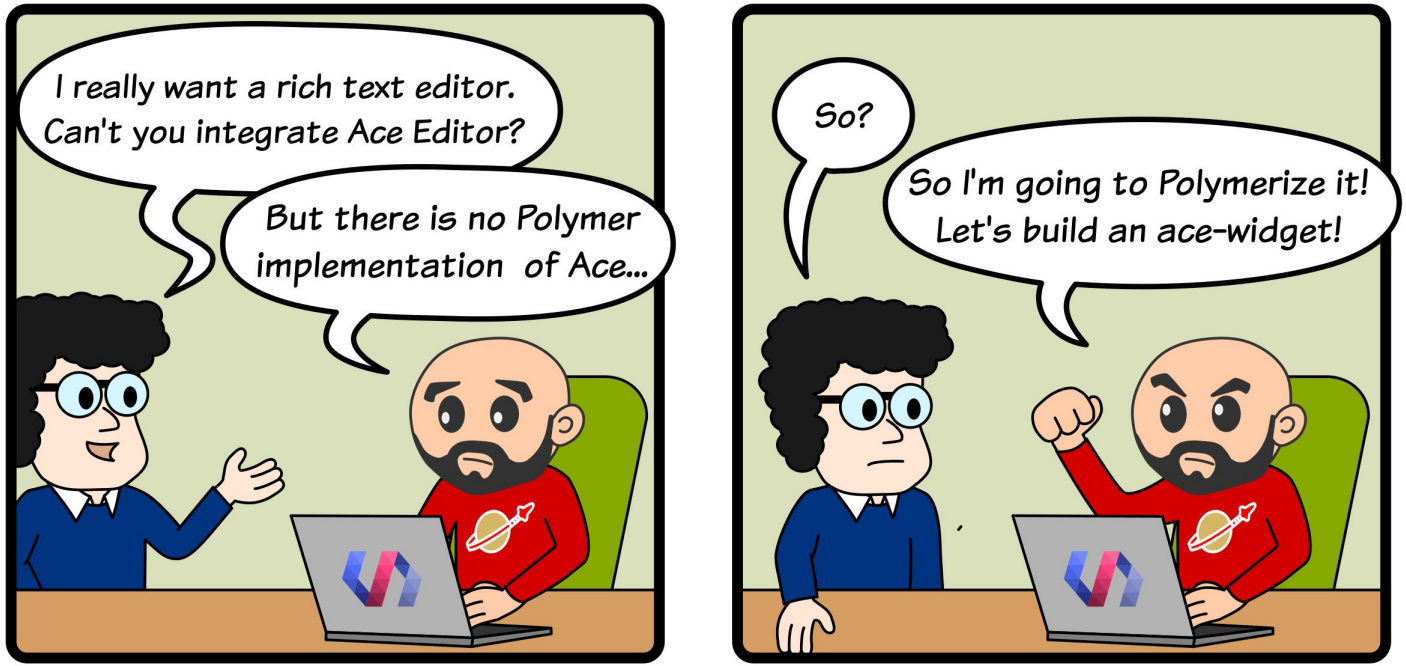
I saw several solutions:

- Wait for the web component
- Dirty integrating the library
- *Componentalize* it

Guess which one I chose...

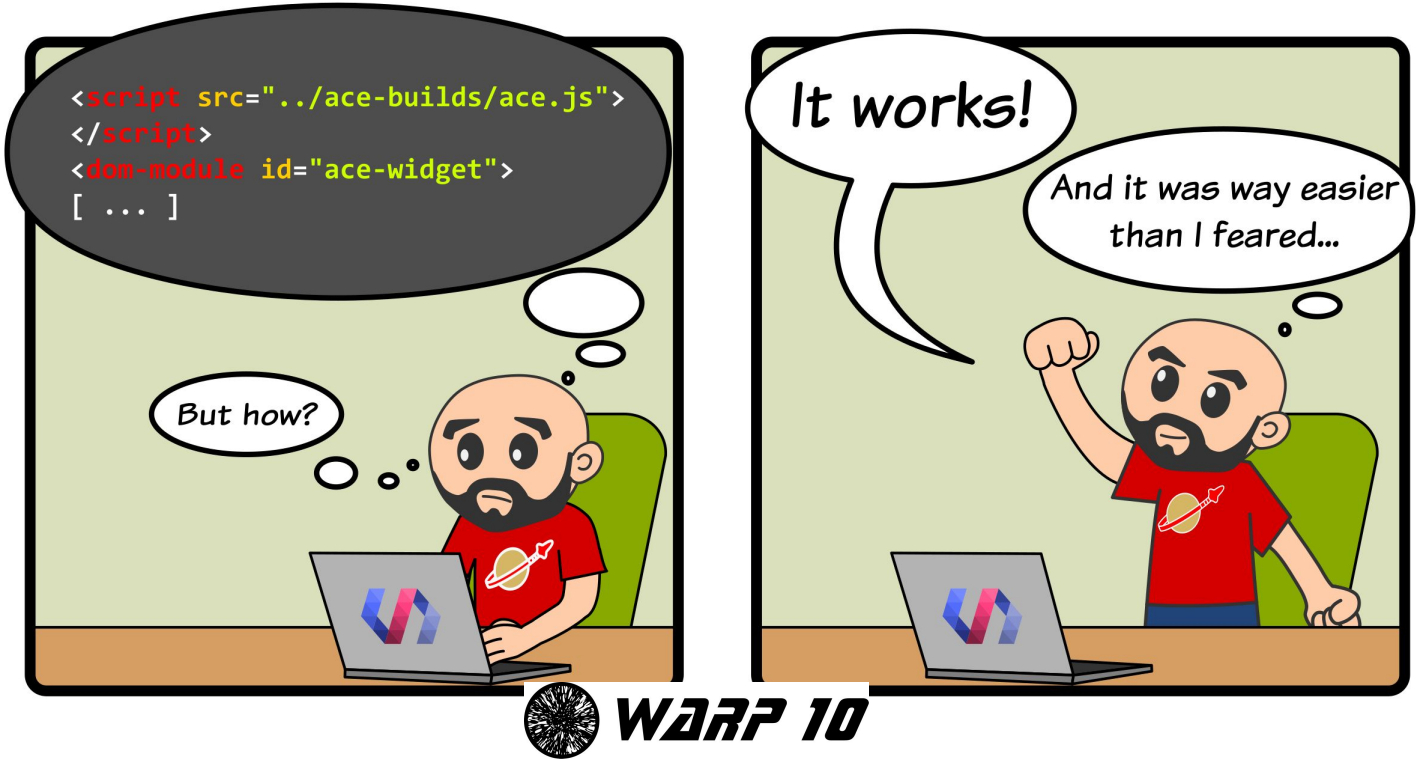


It was only the first time...



WARP 10

How do I *componentalize* them?



WARP 10

Componentalizing a library

Let's begin with a simple example

granite-qrcode-generator

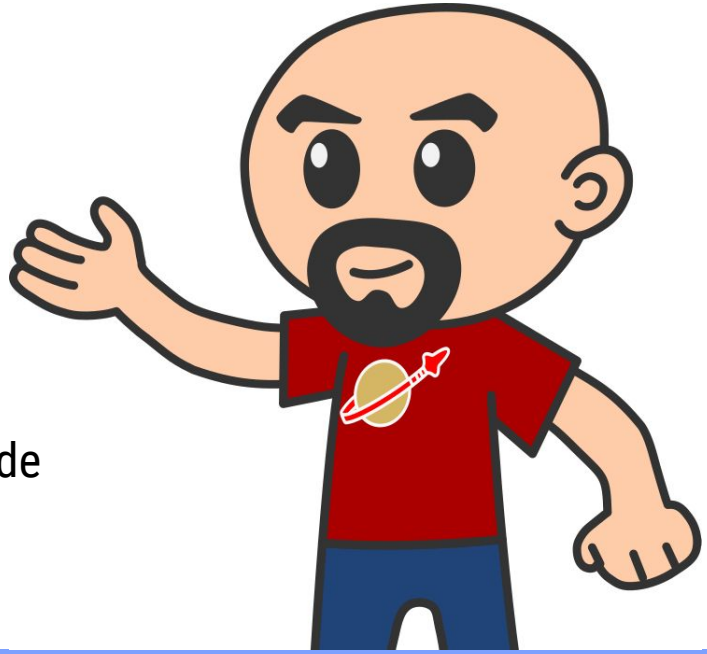
```
<granite-qrcode-generator
    data="https://github.com/lostinbrittany/granite-elements"
    mode="alphanumeric"
    auto></granite-qrcode-generator>
```

What QR Code library to use?

I choose QR.js

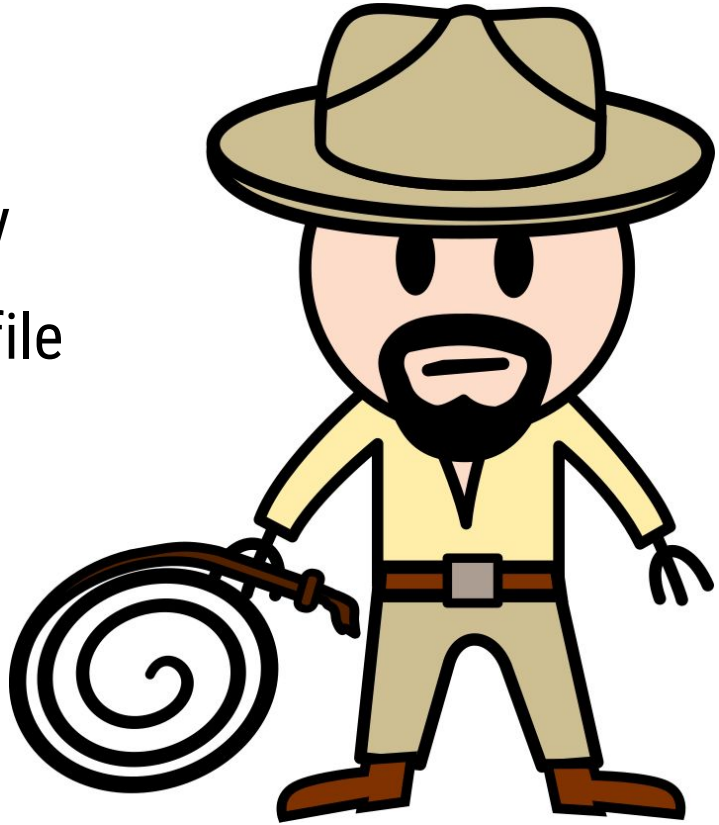
https://github.com/lifthrasiir/qr.js/

- Small
  - 26 kb uncompressed and commented
- Quick!
- Well coded
  - Structured, lots of comments, clean code
- No dirty DOM manipulation



Steps

1. Creating an empty element
2. Add the library as a dependency
3. Load the library in the element file
4. Build a web component encapsulating it
5. Profit?

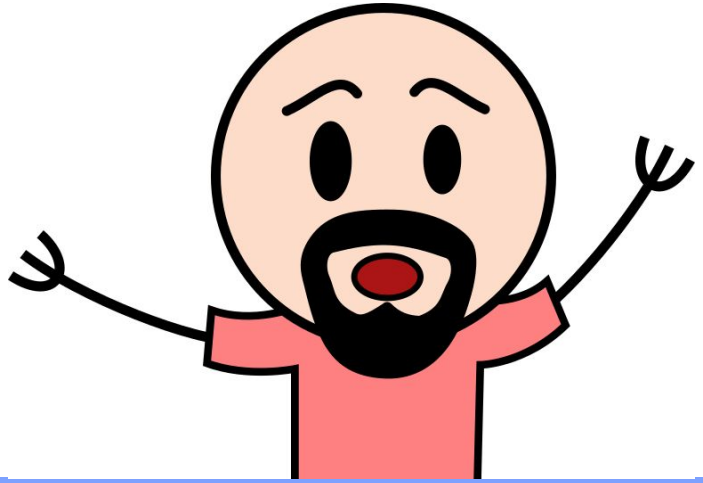


Loading the library in the element file

Usual case: Non-modularized, adding global vars

How to be sure that the lib is

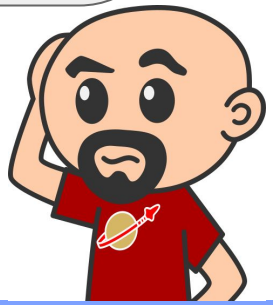
- loaded once
- and only once
- before the element needs it



Loading the library in the element file

```
<script src="../d3/d3.min.js" charset="utf-8"></script>
<script src="../nvd3/build/nv.d3.js"></script>
<!-- include stylesheet for shady dom and shadow dom -->
<link rel="stylesheet" href="../nvd3/build/nv.d3.min.css" />
<link rel="import" type="css" href="../nvd3/build/nv.d3.min.css" />
```

First answer: simply use `script` tag



Loading the library in the element file

2nd answer: Testing and lazy loading
in the element ready lifecycle method…

FOR EVERY ELEMENT
USING A DEP

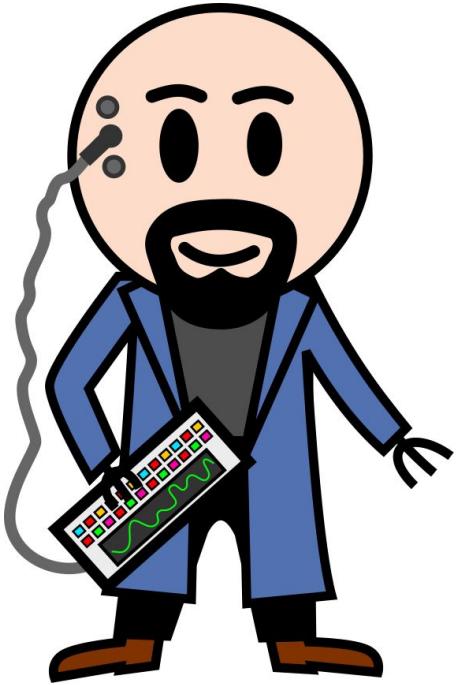
Adding the library as a dependency

3rd answer: *componentalize* the loading!

https://github.com/LostInBrittany/granite-js-dependencies-grabber

```
<link rel="import" href="./granite-c3-css.html">

 <granite-js-dependencies-grabber
     id="granite-js-dependencies-grabber-demo"
     dependencies="[[_dependencies]]"
     on-dependency-is-ready="_onDependencyReady"
     debug="[[debug]]"></granite-js-dependencies-grabber>

_dependencies: { type: Array,
     value: [{name: 'd3', url: '../d3/d3.min.js'},{name: 'c3', url: '../c3/c3.min.js'}] }
```

"Build a web component encapsulating it"

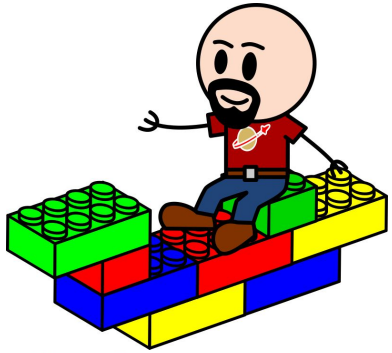


Easier said than done?

1. Define the inputs (attributes)
2. Define the outputs (events)
3. Define the UI (template)
4. Wire the attributes and events to the library
5. Use the lifecycle methods to initialize

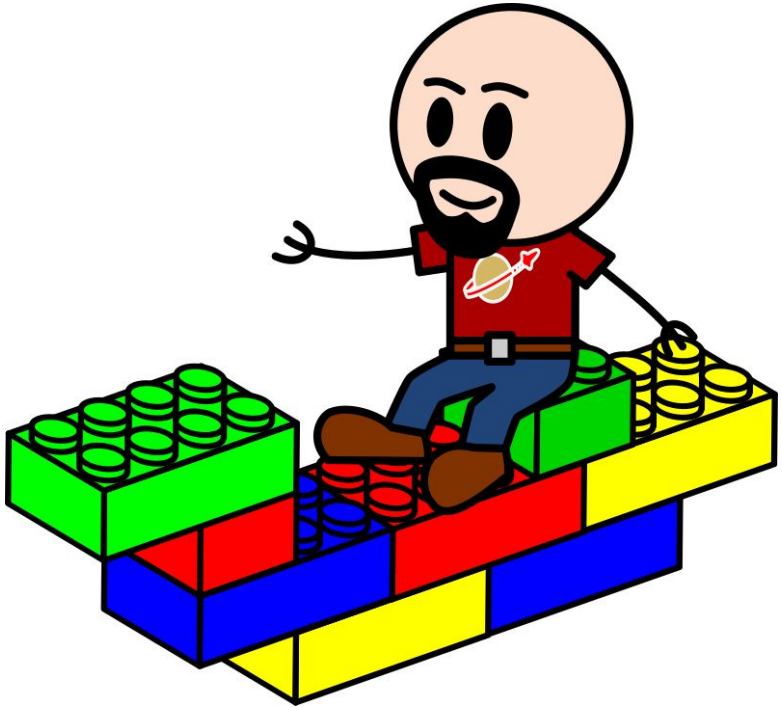
Define the inputs (attributes)

```
50          properties: {
51            /**
52             * The data to encode in the QRCode
53             */
54            data: {
55              type: String,
56            },
57            /**
58              * The format of the generated QRCode, either "html" or "png"
59              * Defaults to "png"
60              */
61            format: {
62              type: String,
63              value: "html"
64            },
65            /**
66             * The size of each modules in pixels
67             * Defaults to 5px
68             */
69            modulesize: {
70              type: Number,
71              value: 5
72            },
73            /**
74             * This is a size of margin in *modules*.
75             * Defaults to 4 (white modules).
76             * The specficiation mandates the margin no less than 4 modules
77             */
78            margin: {
79              type: Number,
80              value: 4
81            },
82            /**
83             * The QRCode version, an integer in [1,40].
84             * When omitted (or -1) the smallest possible version is chosen.
85             */
86            version: {
87              type: Number,
88              value: -1,
89            },
90
91            /**
92             * The mode of the QRCode, one of 'numeric', 'alphanumeric', 'octet'.
93             * When omitted the smallest possible ('numeric') mode is chosen
94             */
95            mode: {
96              type: String,
97              value: "numeric",
98            },
99
100           /**
101            * The error correction code level, one of 'L', 'M', 'Q', 'H'.
102            * Defaults to 'L'.
103            */
104           ecclevel: {
105             type: String,
106             value: 'L',
107           },
108
109           /**
110            * The mask level, an integer in [0,7].
111            * When omitted (or -1) the best mask is chosen
112            */
113           mask: {
114             type: Number,
115             value: -1,
116           },
117
118           /**
119            * If true, the QRCode is regenerated at each change in parameters
120            */
121           auto: {
122             type: Boolean,
123             value: false
124           },
125         },
126
```

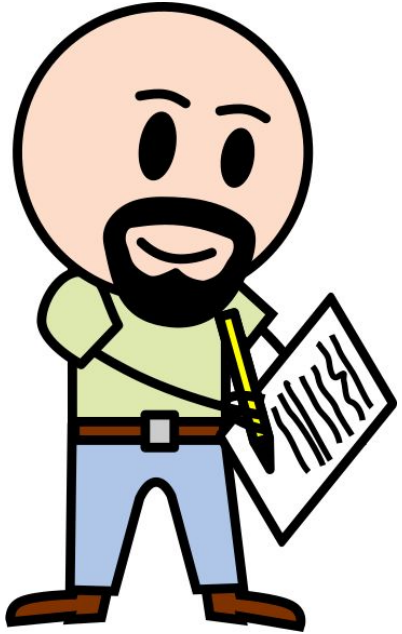


Define the outputs (events)

```
50          /**
51           * Fired when a QR Code is generated.
52           *
53           * @event qrcode-generated
54           */
55
```



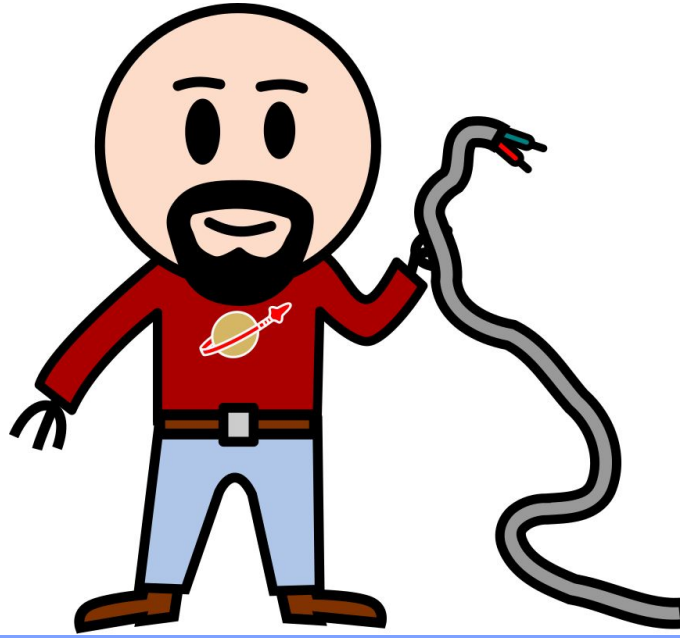
Define the UI (template)



```
36    <template>
37      <style>
38        :host {
39          display: block;
40        }
41      </style>
42      <div id="qrCodeContainer"></div>
43    </template>
44
```

Wire the attributes and events to the library

```
198         _validateParams: function() {
199           return (
200             this._validateModulesize() &&
201             this._validateVersion() &&
202             this._validateMode() &&
203             this._validateMask() &&
204             this._validateEcclevel()
205           );
206         },
207         _validateModulesize: function() {
208           if (this.modulesize >= 0.5) {
209             return true;
210           }
211           console.error("[granite-qrcode-generator] _validateModulesize - Invalid value of `modulesize`", this.modulesize);
212           return false;
213         },
214         _validateMargin: function() {
215           if (this.margin >= -1) {
216             return true;
217           }
218           console.error("[granite-qrcode-generator] _validateMargin - Invalid value of `margin`", this.margin);
219           return false;
220         },
221         _validateVersion: function() {
222           if (this.version == -1 || (this.version >= 0 && this.version <= 40)) {
223             return true;
224           }
225           console.error("[granite-qrcode-generator] _validateVersion - Invalid value of `version`", this.version);
226           return false;
227         },
228         _validateMode: function() {
229           if (this.mode === 'numeric' || this.mode === 'alphanumeric'  || this.mode === 'octet') {
230             return true;
231           }
232           console.error("[granite-qrcode-generator] _validateMode - Invalid value of `mode`", this.mode);
233           return false;
234         },
235         _validateEcclevel: function() {
236           if (this.ecclevel === 'L' || this.ecclevel === 'M'  || this.ecclevel === 'Q' || this.ecclevel === 'H') {
237             return true;
238           }
239           console.error("[granite-qrcode-generator] _validateEcclevel - Invalid value of `ecclevel`", this.ecclevel);
240           return false;
241         },
242         _validateMask: function() {
243           if (this.mask >= -1 && this.mask <=7) {
244             return true;
245           }
246           console.error("[granite-qrcode-generator] _validateMask - Invalid value of `mask`", this.mask);
247           return false;
248         },
```

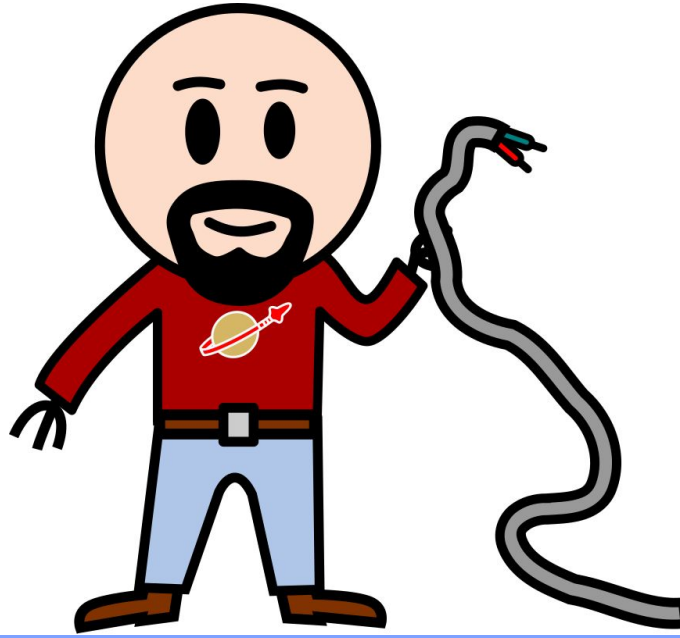


Wire the attributes and events to the library

```
133         observers: [
134           "paramsChanged(data,version,mode,ecclevel,mask,auto)"
135         ],
136
137         // ********************************************************************
138         // Observers
139         // ********************************************************************
140         paramsChanged: function() {
141           console.debug("[granite-qrcode-generator] paramsChanged - auto ", this.auto);
142           if (this.auto) {
143             this.generateQRCode();
144           }
145         },
146
```

Wire the attributes and events to the library

```
151         /**
152          * Generates the QRCode
153          */
154         generateQRCode: function() {
155           if (!this._validateParams()) {
156             return;
157           }
158           var options = {
159             modulesize: this.modulesize,
160             margin: this.margin,
161             version: this.version,
162             mode: this.mode,
163             ecclevel: this.ecclevel,
164             mask: this.mask
165           }
166           if (this.format === 'png') {
167             this.generateQRCodePNG(options);
168           }
169           else {
170             this.generateQRCodeHTML(options);
171           }
172           this.fire("qrcode-generated");
173         },
174         generateQRCodePNG: function (options) {
175           var img;
176           try {
177             img = document.createElement('img');
178             img.src = QRCode.generatePNG(this.data, options);
179             this._appendQRCode(div);
180           }
181           catch (e) {
182             console.log('no canvas support');
183           }
184         },
185         generateQRCodeHTML: function (options) {
186           console.debug("[granite-qrcode-generator] generateQRCodeHTML - data ", this.data);
187           var div = QRCode.generateHTML(this.data, options);
188           this._appendQRCode(div);
189         },
190
191         _appendQRCode: function(node) {
192           for (var i=Polymer.dom(this.$.qrCodeContainer).children.length-1; i>=0 ; i--) {
193                Polymer.dom(this.$.qrCodeContainer).removeChild( Polymer.dom(this.$.qrCodeContainer).children[i]);
194           }
195           Polymer.dom(this.$.qrCodeContainer).appendChild(node);
196         },
197
```



granite-qrcode-generator

```
<granite-qrcode-generator
    data="https://github.com/lostinbrittany/granite-elements"
    mode="alphanumeric"
    auto></granite-qrcode-generator>
```

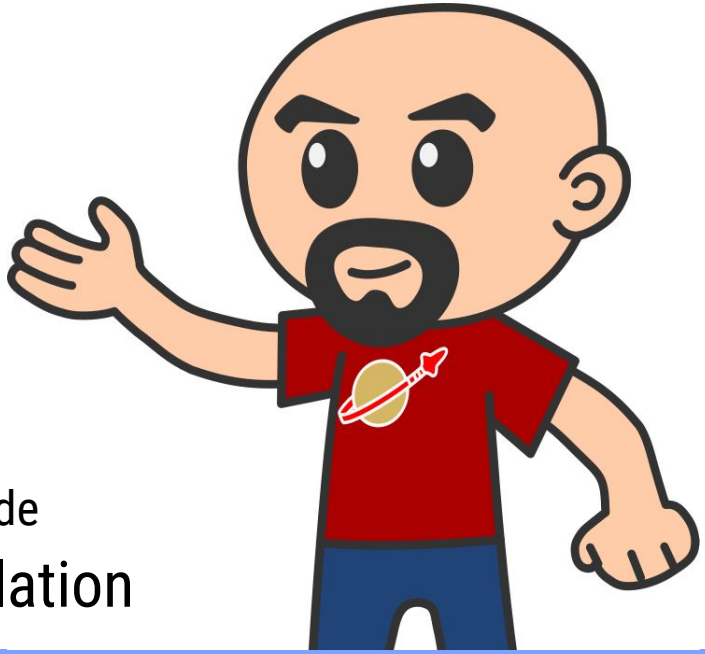
granite-qrcode-scanner

What QR Code scan library to use?

I choose jsqrcode

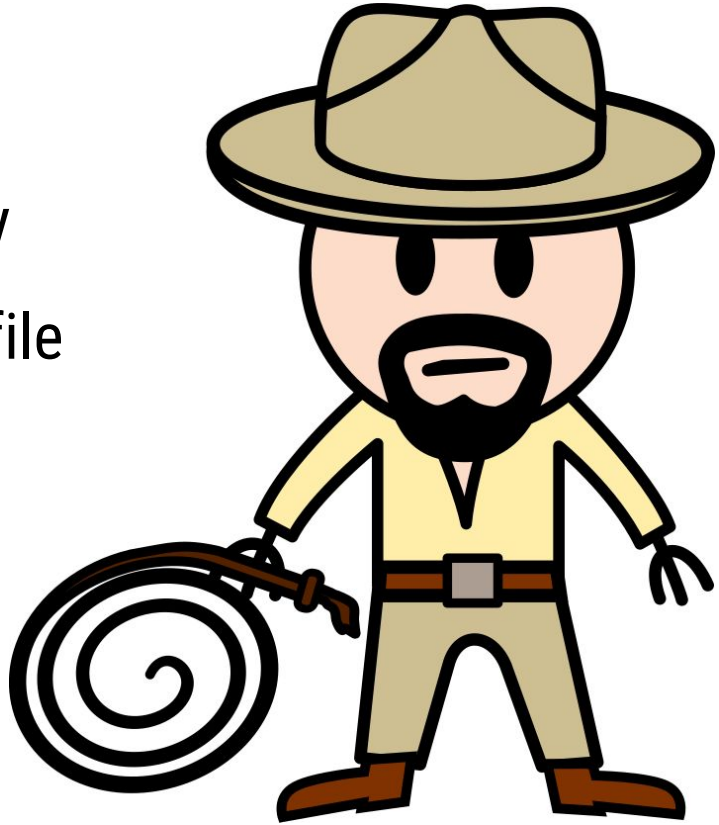
https://github.com/LazarSoft/jsqrcode

- Small for a full QR Code scanner
  - 110 kb uncompressed and commented
- Quick and efficient
- Well coded
  - Structured, lots of comments, clean code
- But with some dirty DOM manipulation

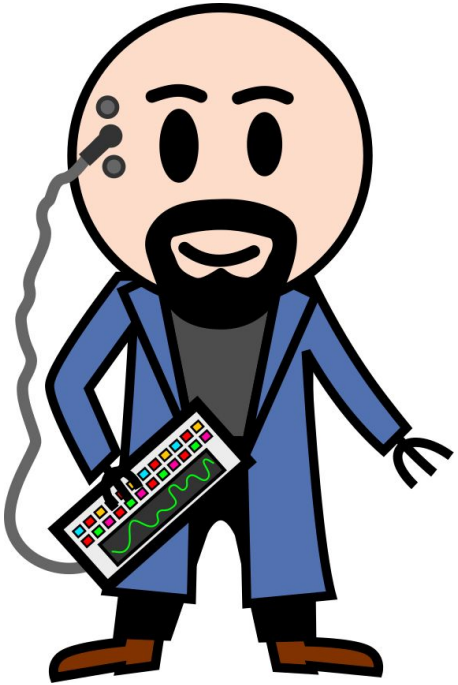


Steps

1. Creating an empty element
2. Add the library as a dependency
3. Load the library in the element file
4. Build a web component encapsulating it
5. Profit?



"Build a web component encapsulating it"

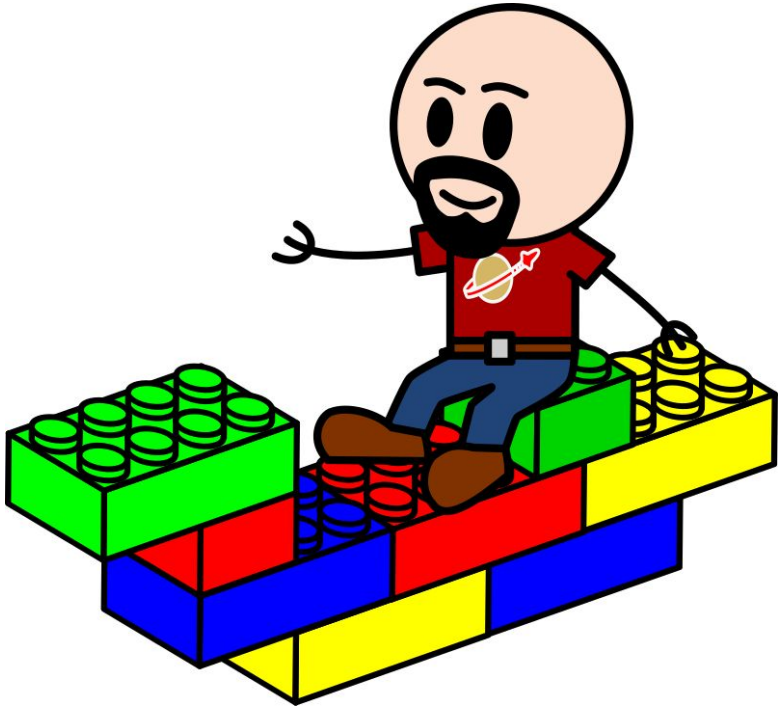


Easier said than done?

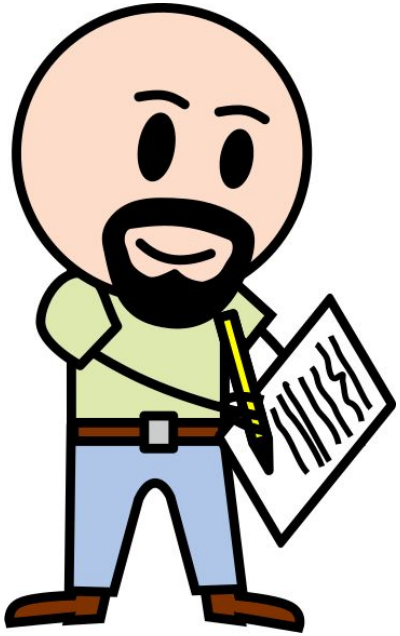
1. Define the inputs (attributes)
2. Define the outputs (events)
3. Define the UI (template)
4. Wire the attributes and events to the library
5. Use the lifecycle methods to initialize

Define the inputs and outputs

```
70          properties: {
71            /**
72              * If true the elements scans for QR code
73              */
74            active: {
75              type: Boolean,
76              value: false
77            },
78
79            /**
80             * The last decoded QRCode
81             */
82            data: {
83              type: String,
84              notify: true,
85              value: "",
86            },
87            /**
88             * The width of the scanning window
89             */
90            width: {
91              type: Number,
92              value: 320
93            },
94            /**
95             * The height of the scanning window
96             */
97            height: {
98              type: Number,
99              value: 240
100           },
```



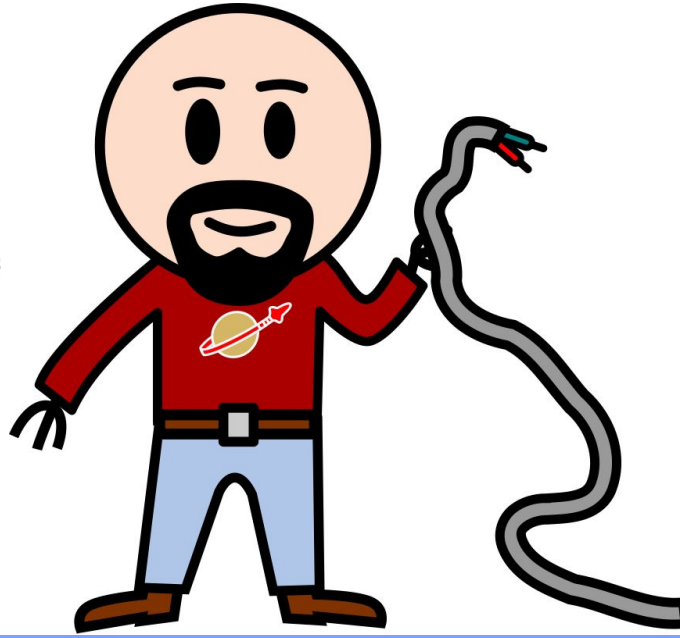
Define the UI (template)



```
33      <template>
34        <style>
35          :host {
36            display: block;
37          }
38          [hide] {
39            display: none;
40          }
41          .media {
42              display: flex;
43              flex-flow: column nowrap;
44              align-items: center;
45          }
46        </style>
47        <div class="media">
48            <video id="qrVideo" autoplay width="[[width]]" height="[[height]]" hide$="[[!_supportsWebRtc]]"></video>
49          <template is="dom-if" if="[[!_supportsWebRtc]]">
50            <granite-file-reader
51                read-as="dataURL"
52                accept=".jpg"
53                on-file-read="_onFileRead">
54              <div>
55                <svg xmlns="http://www.w3.org/2000/svg" width="256" height="256" viewBox="0 0 256 256"><path d="M19.6 3.9C10.8
                  19.6 252.3L236.2 252.3C245 252.3 252.1 245.2 252.1 236.4L252.1 19.8C252.1 11 245 3.9 236.2 3.9L19.6 3.9zM108.8
                  60.4 165.5 62.5 164.7 64.6L204.1 64.6C216.8 64.6 229.5 77.3 229.5 90L229.5 191.6C229.5 204.3 216.8 217 204.1 2
                  90C26.3 77.3 39 64.6 51.7 64.6L91.1 64.6C90.3 62.5 89.8 60.4 89.8 58.2 89.8 48.7 99.3 39.2 108.8 39.2zM127.9 9
                  191.6 127.9 191.6 156 191.6 178.7 168.9 178.7 140.8 178.7 112.7 156 90 127.9 90zM127.9 115.4C141.9 115.4 153.3
                  166.2 113.9 166.2 102.5 154.8 102.5 140.8 102.5 126.8 113.9 115.4 127.9 115.4z" style="fill:#ffc107;stroke-
56              </div>
57            </granite-file-reader>
58          </template>
59
60
61          <canvas id="qrCanvas" width="[[_canvasWidth]]" height="[[_canvasHeight]]" hide></canvas>
62        </div>
63      </template>
64
```

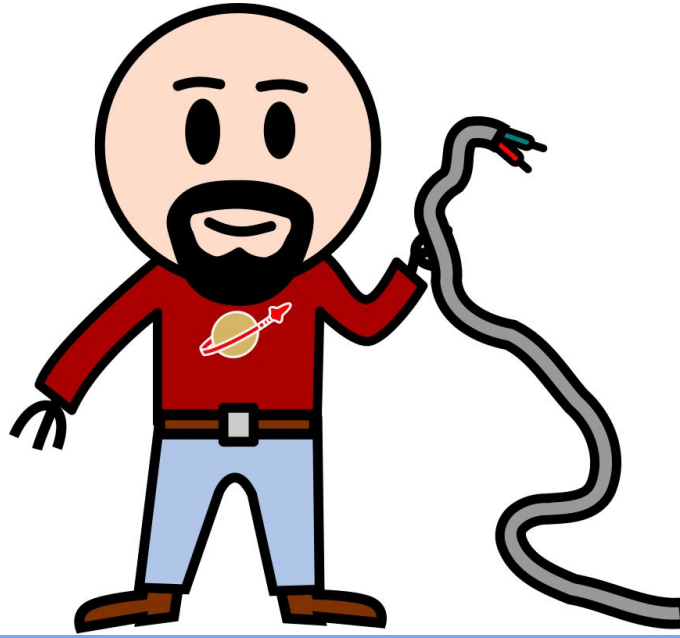
Initializing in the lifecycle methods

```
130
131         // ********************************************************************
132         // Livecycle
133         // ********************************************************************
134         attached: function() {
135
136           this._supportsWebRtc = this._doesSupportWebRtc();
137
138           this._context = this.$.qrCanvas.getContext("2d");
139
140           this._context.clearRect(0, 0,  this._canvasWidth, this._canvasHeight);
141
142           var elem = this;
143           //called when qrcode is found
144           qrcode.callback = function(res) {
145             elem.data = res;
146             console.debug("[granite-qrcode-scanner] qrcode.callback", elem.data, elem);
147           };
148
149
150           if (this._supportsWebRtc) {
151             this._initWebcam();
152           }
153
154
155         },
```



Initializing in the lifecycle methods

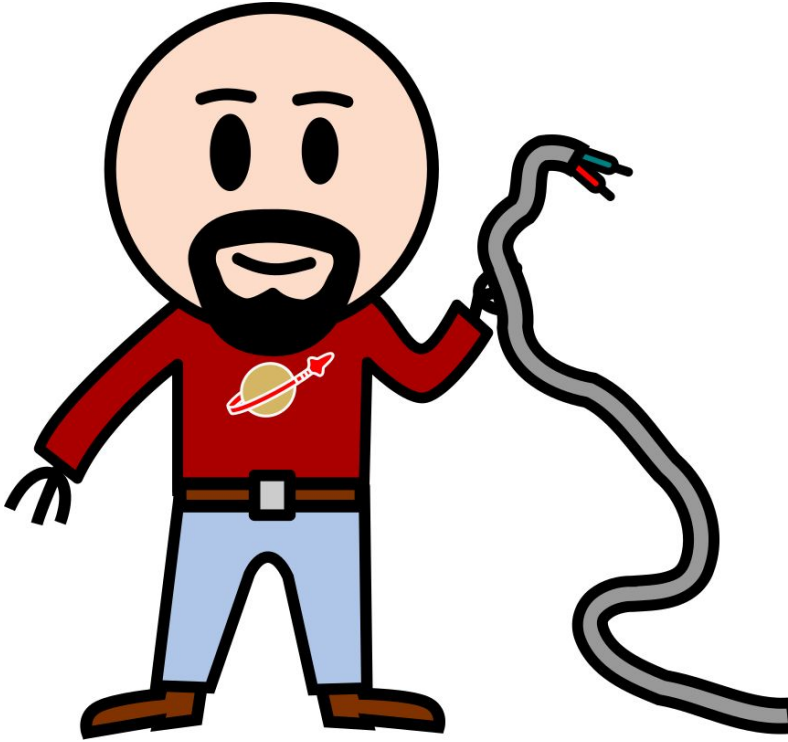
```
213         _initWebcam: function() {
214           var options = true;
215           var elem = this;
216           if(navigator.mediaDevices && navigator.mediaDevices.enumerateDevices)   {
217             try{
218               navigator.mediaDevices.enumerateDevices()
219               .then(function(devices) {
220                 devices.forEach(function(device) {
221                   if (device.kind === 'videoinput') {
222                     console.debug("[granite-qrcode-scanner] _initWebcam - device found", device.kind + ": " + device.label +" id = " + device.deviceId);
223                     if(device.label.toLowerCase().search("back") >-1 /* || device.label.toLowerCase().search("rear") >-1 */) {
224                       options={'deviceId': {'exact':device.deviceId}, 'facingMode': 'environment'} ;
225                     }
226                   }
227                   console.debug("[granite-qrcode-scanner] _initWebcam", device.kind + ": " + device.label +" id = " + device.deviceId, "options", options);
228                 });
229                 elem._initWebcam2(options);
230               });
231             }
232             catch(e)
233             {
234               console.log("[granite-qrcode-scanner] _initWebcam - error", e);
235             }
236           }
237           else{
238             console.debug("[granite-qrcode-scanner] _initWebcam - no navigator.mediaDevices.enumerateDevices" );
239             elem._initWebcam2(options);
240           }
241         },
242
243         _initWebcam2: function(options) {
244
245             console.debug("granite-qrcode-scanner] _initWebcam2",options);
246             if(this._stype==1) {
247                 this.async(this._captureVideo, this._refresh);
248                 return;
249             }
250
251             var elem = this;
252             var moz, webkit
253             //webcam activation
254             if (navigator.getUserMedia) {
255               navigator.getUserMedia({
256                   video: options, audio: false,
257                 }, _onCameraSuccess, _onCameraError);
258             } else if (navigator.webkitGetUserMedia) {
259               this._browser = "webkit";
260               navigator.webkitGetUserMedia({video: options, audio: false}, _onCameraSuccess, _onCameraError);
261             } else if(navigator.mediaDevices && navigator.mediaDevices.getUserMedia) {
262               this._browser = "moz";
263               navigator.mediaDevices.
264                   getUserMedia({video: options, audio: false}).
265                   then(_onCameraSuccess).catch(_onCameraError);
266             } else if(navigator.mozGetUserMedia) {
267               this._browser = "moz";
268               navigator.mozGetUserMedia({video: options, audio: false}, _onCameraSuccess, _onCameraError);
269             }
270
271             function _onCameraSuccess(stream) {
272               if(elem._browser == "webkit") {
273                 elem.$.qrVideo.src = window.webkitURL.createObjectURL(stream);
274               } else if(elem._browser == "moz") {
275                 elem.$.qrVideo.mozSrcObject = stream;
276                 elem.$.qrVideo.play();
277               } else {
278                 elem.$.qrVideo.srcObject = stream;
279               }
280               elem._captureVideo();
281             }
282             function _onCameraError(e) {
283               console.log("[granite-qrcode-scanner] _onCameraError", e);
284               alert("Can't access to webcam");
285             }
286
287             this._stype=1;
288             this.async(this._captureVideo, this._refresh);
289         },
290
```



But what about the wiring?

Almost no wiring needed

- Either done in the template
- Or in the initialization

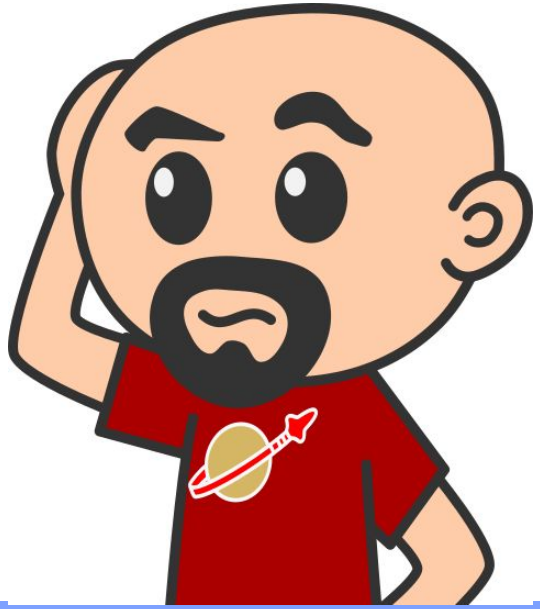


And then, does it work?

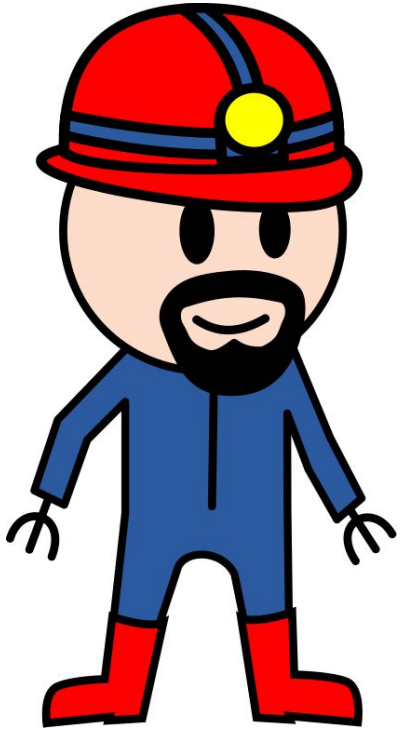
Weeeeell, not really…

And it doesn't give a clear error

What does it happen here ?



Digging in the problem



Going deep inside the library

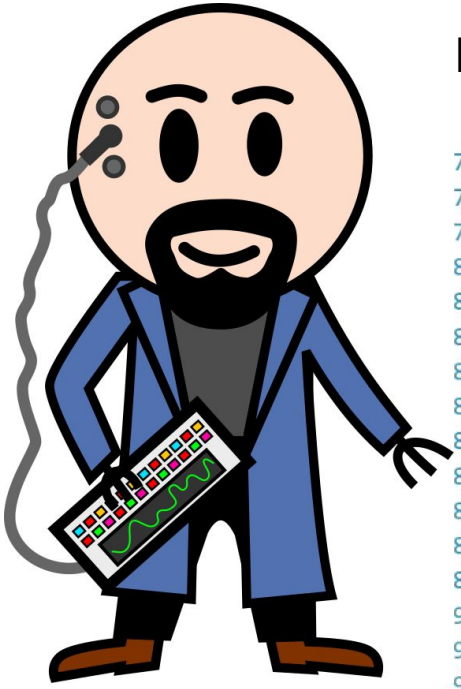
Adding logs and breakpoints

And I found the guilty line:

https://github.com/LazarSoft/jsqrcode/blob/master/src/qrcode.js

```
qrcode.height = 0;
qrcode.qrCodeSymbol = null;
qrcode.debug = false;
qrcode.maxImgSize = 1024*1024;

qrcode.sizeOfDataLengthInfo =  [  [ 10, 9, 8, 8 ],  [ 12, 11, 16, 10 ],  [ 14, 13, 16, 12 ] ];

qrcode.callback = null;

qrcode.decode = function(src){

    if(arguments.length==0)
    {
        var canvas_qr = document.getElementById("qr-canvas");
        var context = canvas_qr.getContext('2d');
        qrcode.width = canvas_qr.width;
        qrcode.height = canvas_qr.height;
        qrcode.imagedata = context.getImageData(0, 0, qrcode.width, qrcode.height);
        qrcode.result = qrcode.process(context);
        if(qrcode.callback!=null)
            qrcode.callback(qrcode.result);
        return qrcode.result;
    }
```

Patching the library



Doing it the open source way...

```
77      var canvas_qr;
78      if(arguments.length==0) {
79        canvas_qr = document.getElementById("qr-canvas");
80      } else {
81        canvas_qr = qrCanvas;
82      }
83      var context = canvas_qr.getContext('2d');
84      qrcode.width = canvas_qr.width;
85      qrcode.height = canvas_qr.height;
86      qrcode.imagedata = context.getImageData(0, 0, qrcode.width, qrcode.height);
87      qrcode.result = qrcode.process(context);
88      if(qrcode.callback!=null) {
89        qrcode.callback(qrcode.result);
90      }
91      return qrcode.result;
92    }
```

granite-qrcode-scanner

Other examples: ace-widget

```
1  Write something... Anything...
```

```
<ace-widget placeholder="Write something... Anything..." initial-focus>
</ace-widget>
```

Thanks!

I hope you liked this talk!

Don't hesitate to send me your questions
by email, twitter, hangout, carrier pigeon...