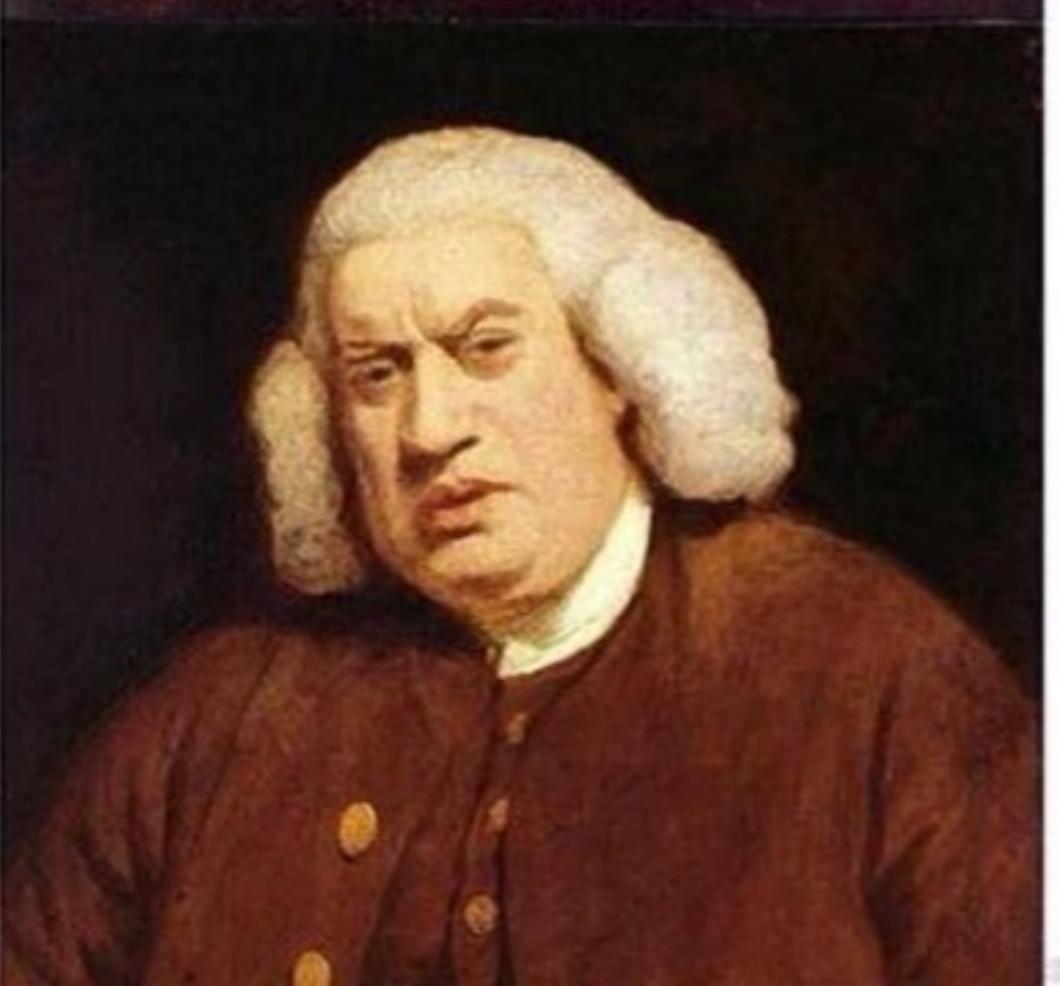
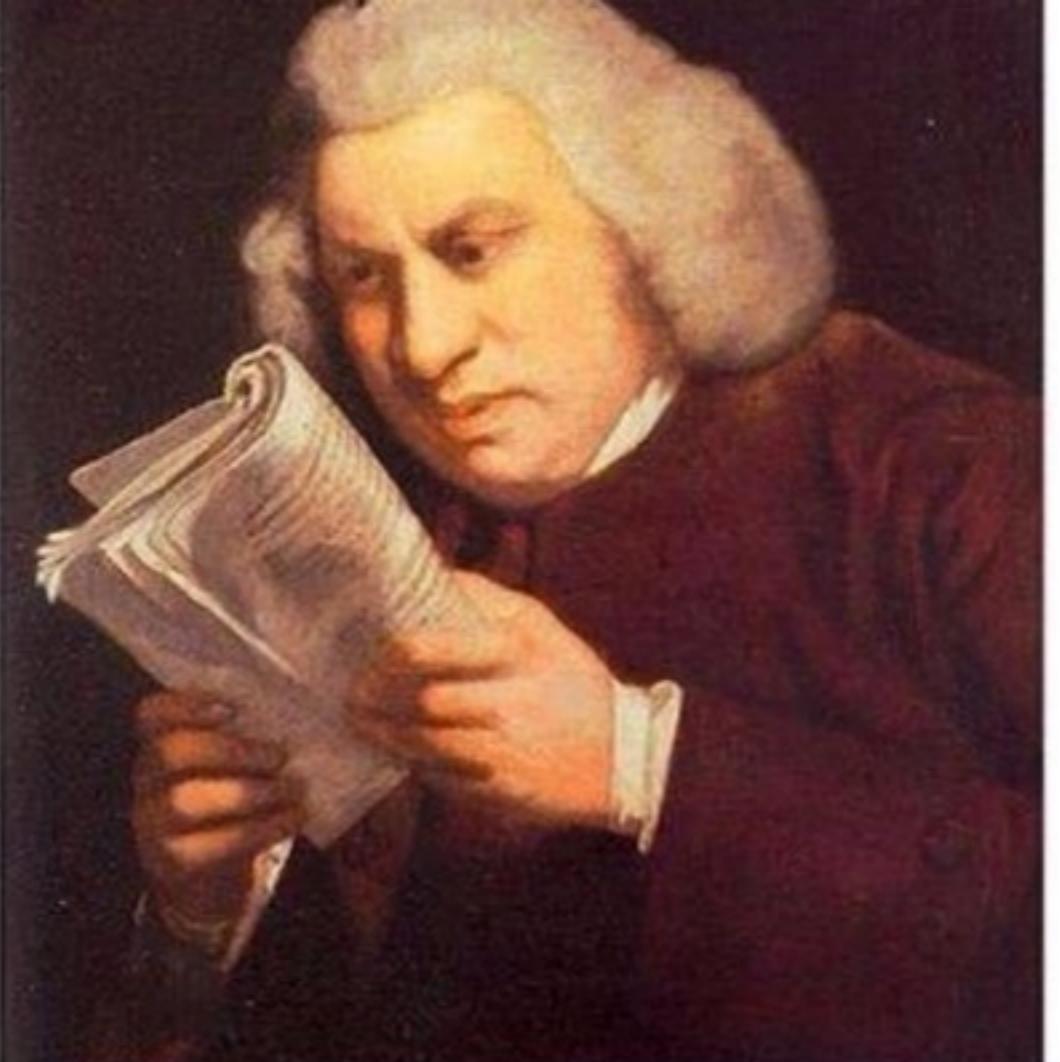


ЖУКОВА ЕЛЕНА

СВЕТЛОЕ БУДУЩЕЕ CUSTOM
ELEMENTS



О ЧЕМ ЭТО?

ДМИТРИЙ ДИМАНД



DOM медленный



JSX лучше чем DOM API



Строчные атрибуты - отстой

Ты медленный



JSX - это просто сахар

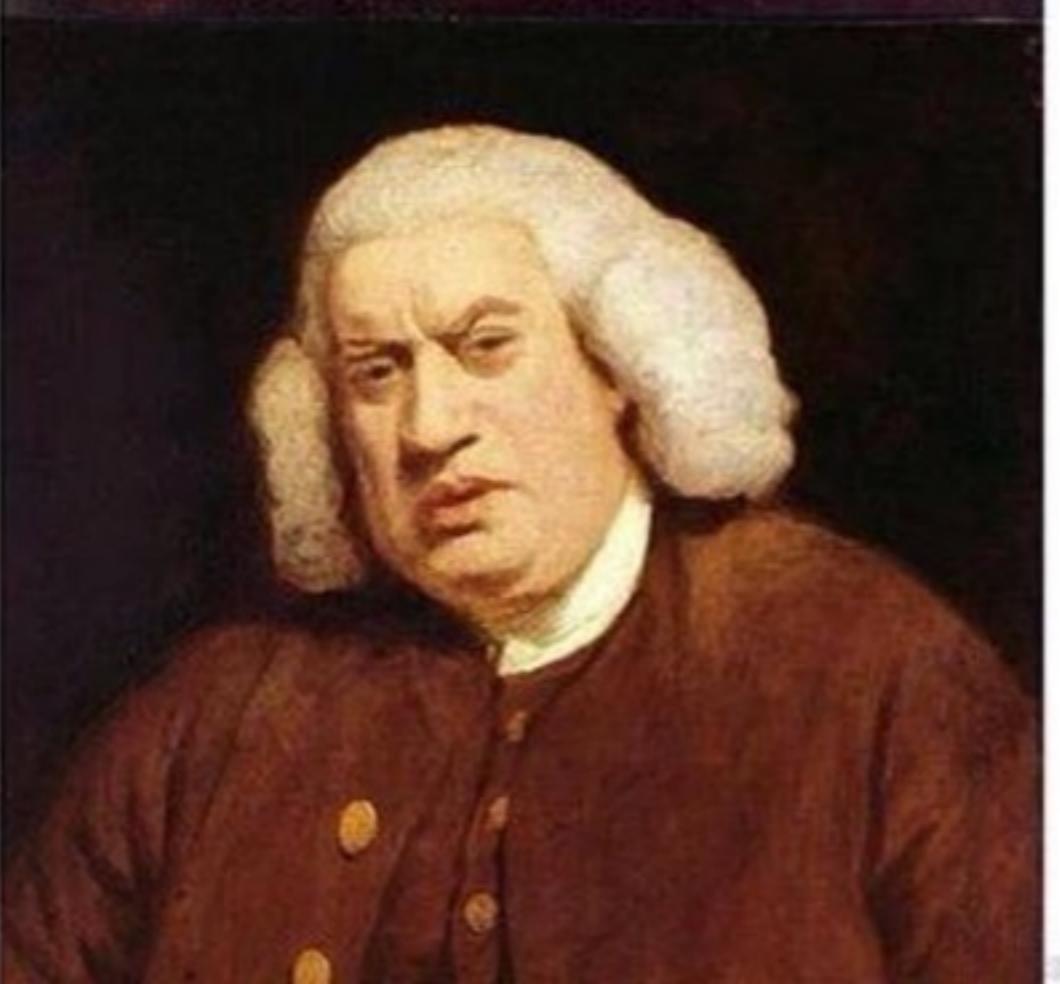
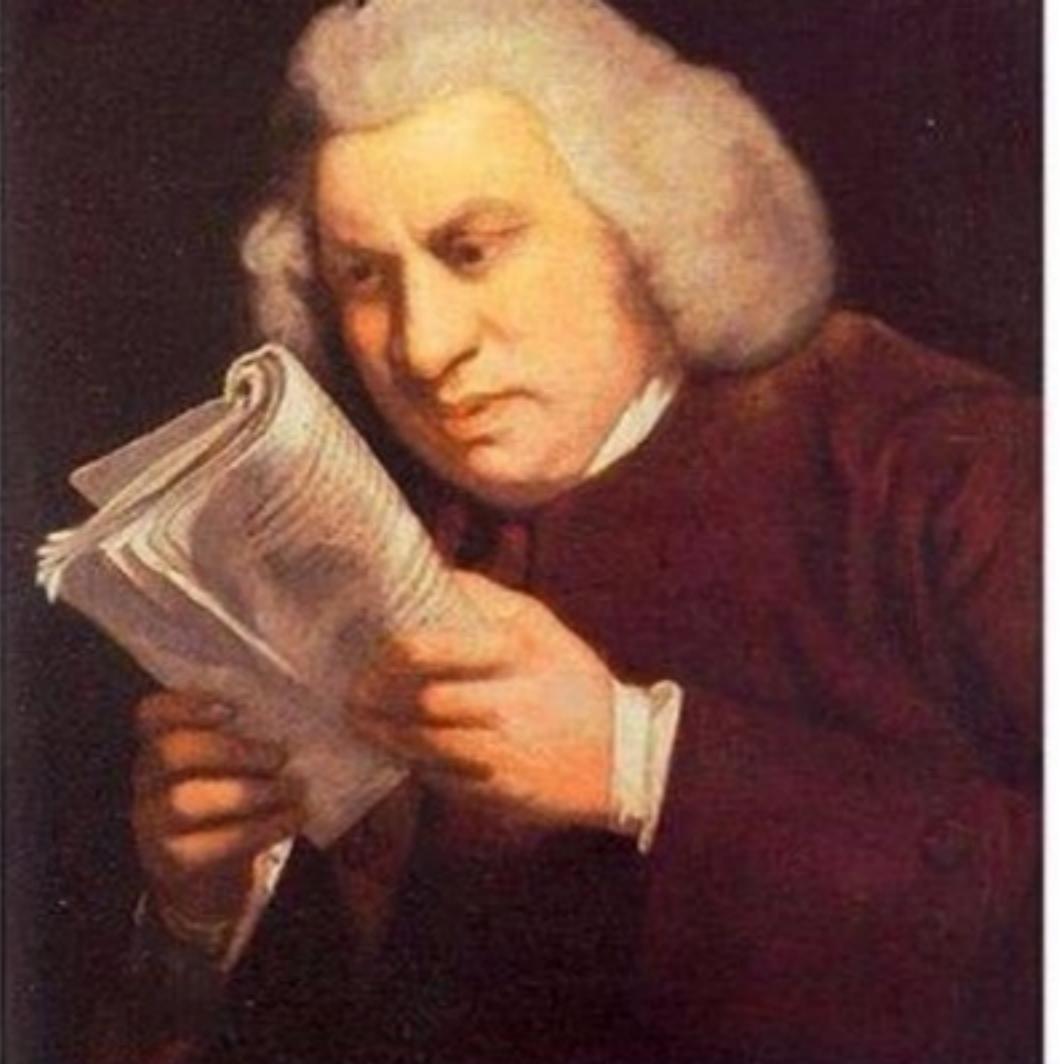


Используй атрибуты



* Нарушенное обещание Web Components (The broken promise of Web Components)

** О нарушенном обещании Web Components (Regarding the broken promise of Web Components)



ИСТОРИЯ

РОЖДЕНИЕ

- ▶ Игрушечный язык
- ▶ Язык-прокладка для дизайнеров и программистов-любителей
- ▶ Писать на JavaScript значило быть не серьезным программистом



МОНСТР?

- ▶ Совместимость в браузерах
- ▶ Манипуляция DOM
- ▶ Отсутствие объектно-ориентированного стиля
- ▶ Проблемы с масштабированием
- ▶ Сложная поддержка



ОРУЖИЕ

- ▶ jQuery
- ▶ Backbone
- ▶ Knockout
- ▶ Ember
- ▶ Angular
- ▶ React
- ▶ Vue

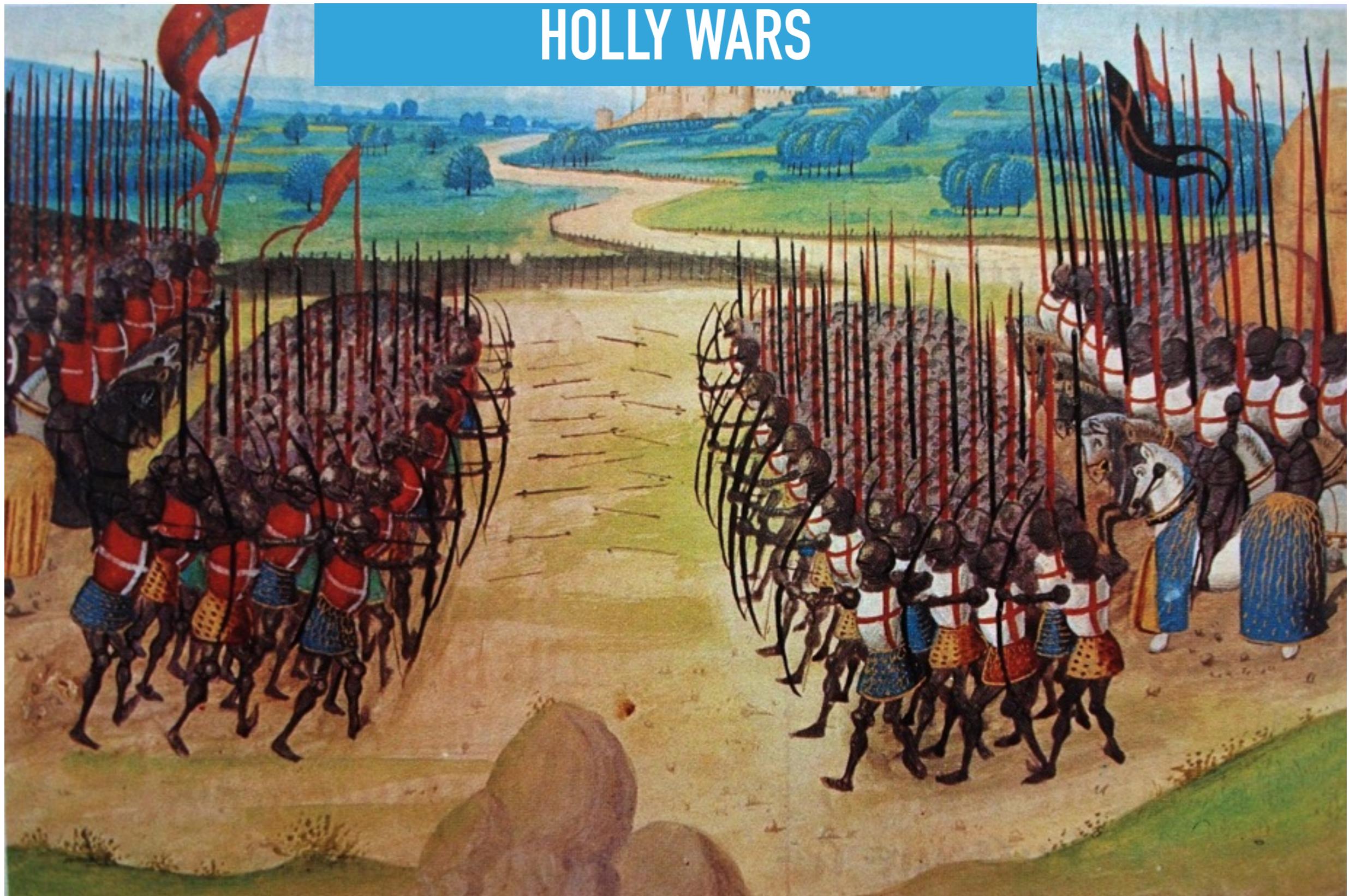


ДОСПЕХИ

- ▶ Webpack
- ▶ Browserify
- ▶ Babel
- ▶ Gulp
- ▶ и еще 100500 других

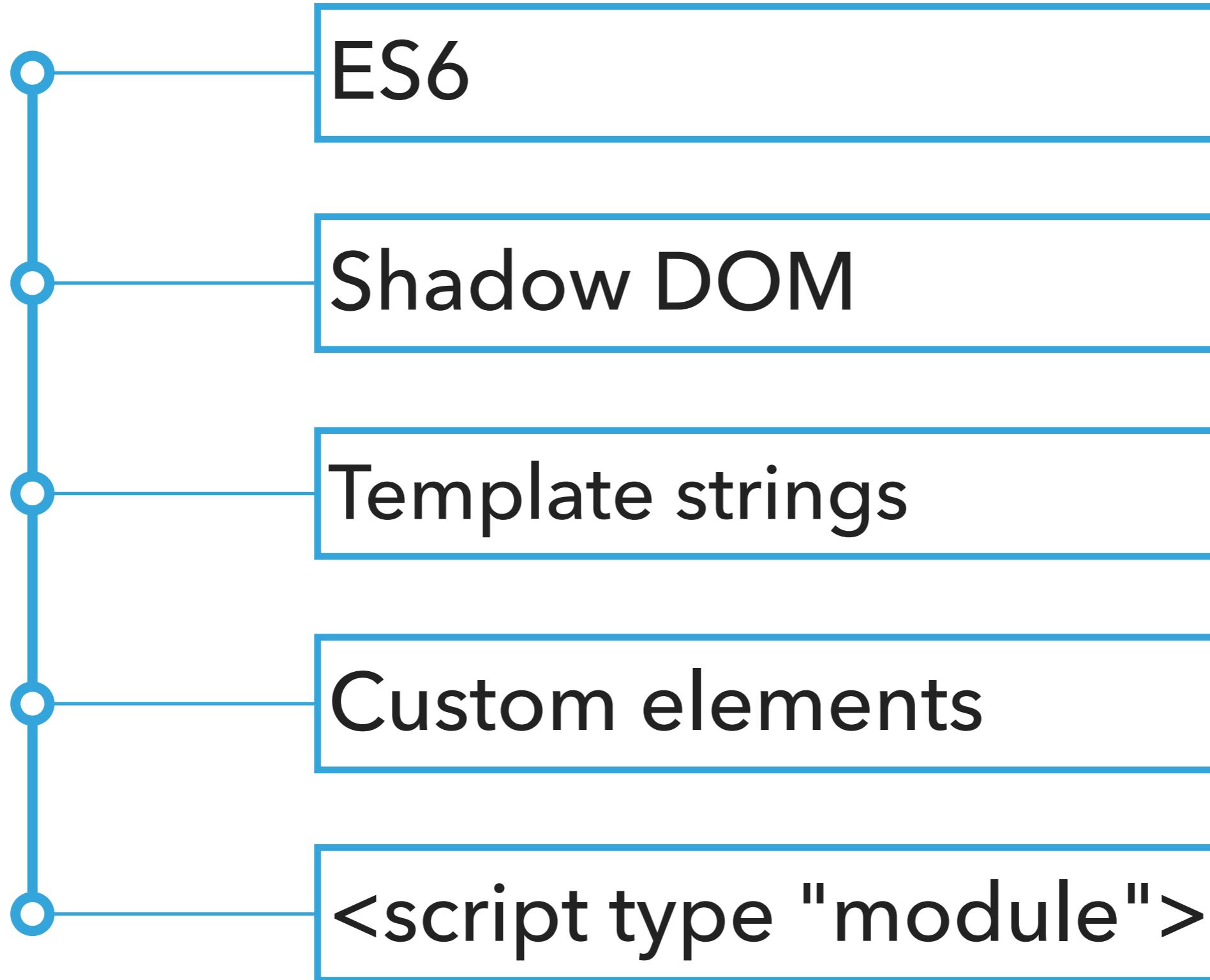


HOLLY WARS

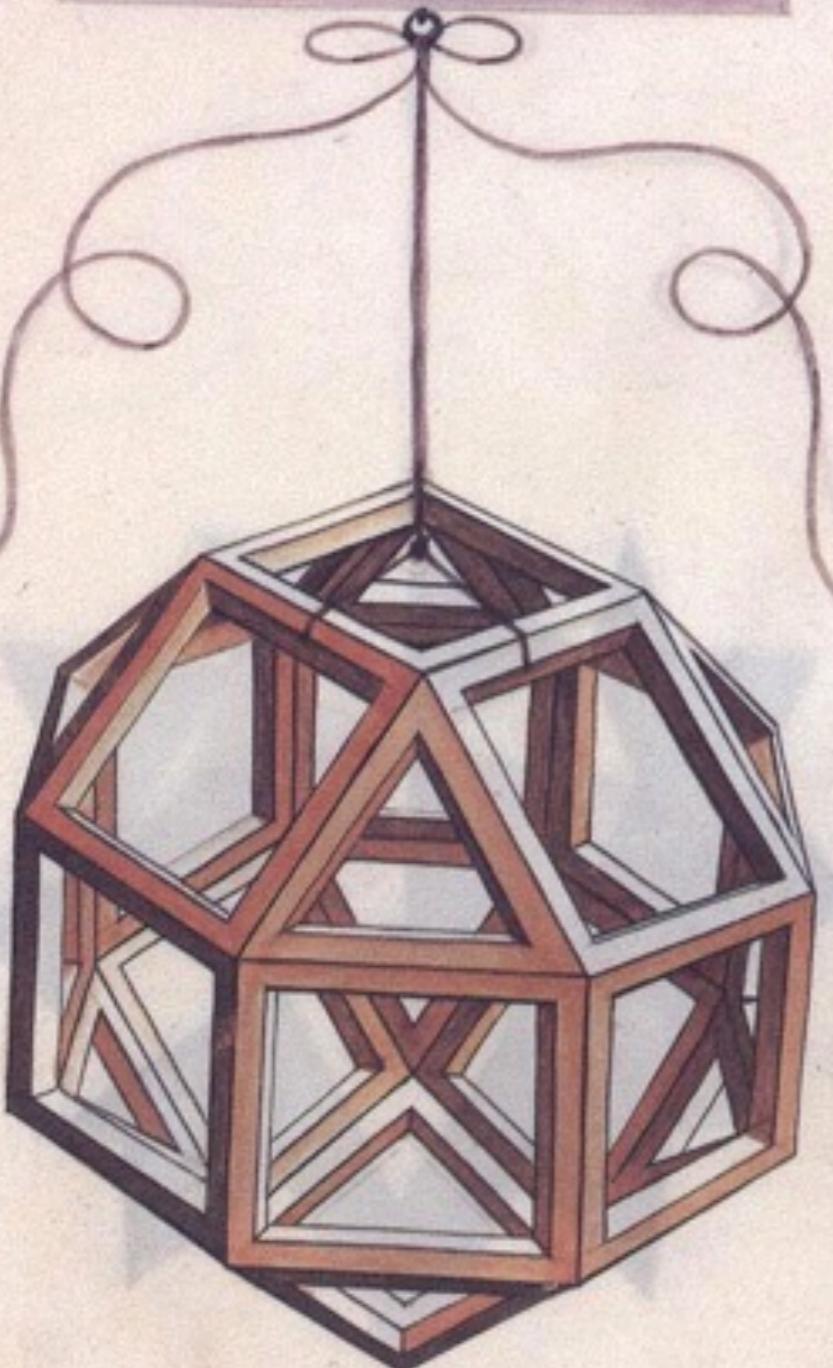




СТАНДАРТЫ



VIGINTISEX BASIVM
PLANVS VACVVS



CUSTOM ELEMENTS

CUSTOM ELEMENTS

WEB COMPONENTS AND MODEL DRIVEN VIEWS 2011



Look at my kitten isn't he quuuute

x-comment | 0 x 0

COMMENTS
ADD COMMENT

Post

ALEX RUSSEL



Elements Memory Console Sources Audits Network Performance Application

```
<!-- Template for the Comment element -->
<template id="commentTemplate">..</template>
<script>
  HTMLElement.call = Function.prototype.call; // Baling wire--this will go away
  getById = function(id) { return document.getElementById(id); }

  function Comment(text) {
    HTMLElement.call(this); // Makes this an Element
    this.textContent = text || getById("lorem").innerText;
    this.shadow = new ShadowRoot(this); // Give us a Shadow
    this.buildUI();
  }

  Comment.prototype = Object.create(HTMLElement.prototype);
  Comment.prototype.constructor = Comment;

  Comment.prototype.buildUI = function() {
    var holder = getById('commentTemplate').firstElementChild.cloneNode(true);
    this.shadow.appendChild(holder);
    holder.querySelector('.avatar').src = 'avatar' + Math.floor(Math.random() * 5)
    + '.jpg';
  }

  HTMLElement.register('x-comment', Comment);

  function postClicked() {
    var textarea = document.querySelector('textarea');
    post(textarea.value);
    textarea.value = '';
  }

  function post(text) {
    getById("comments").appendChild(
      new Comment(text)
    );
  }

</script>
<h2>Comments</h2>
<div id="comments">
  <x-comment>
  </x-comment> == $0
</div>
<h2>Add Comment</h2>
<textarea></textarea>

```

html body div#comments x-comment

margin border padding auto x auto 13.333

CUSTOM ELEMENTS

CUSTOM ELEMENTS ИСТОРИЯ ПУБЛИКАЦИЙ

2016-10-13	Working Draft
2016-10-02	Working Draft
2016-08-30	Working Draft
2016-07-21	Working Draft
2016-06-21	Working Draft
2016-06-09	Working Draft
2016-05-23	Working Draft
2016-05-17	Working Draft
2016-02-26	Working Draft
2016-01-19	Working Draft
2014-12-16	Working Draft
2013-10-24	Last Call
2013-05-14	First Public Draft

CUSTOM ELEMENTS 2013



The screenshot shows a code editor window titled "old-way.js — ~/Documents/custom elements". The file contains 17 numbered lines of JavaScript code. Lines 1 through 16 are highlighted in light gray, while line 17 is white. The code demonstrates an older, manual approach to creating a custom element. It starts by creating a new object based on the HTML Element prototype (line 3). It then defines four callback properties: createdCallback (line 5), attachedCallback (line 7), detachedCallback (line 9), and attributeChangedCallback (line 11). Finally, it registers the custom element with the document (line 13).

```
old-way.js
1 (function() {
2     // Creates an object based in the HTML Element prototype
3     var element = Object.create(HTMLElement.prototype);
4     // Fires when an instance of the element is created
5     element.createdCallback = function() {};
6     // Fires when an instance was inserted into the document
7     element.attachedCallback = function() {};
8     // Fires when an instance was removed from the document
9     element.detachedCallback = function() {};
10    // Fires when an attribute was added, removed, or updated
11    element.attributeChangedCallback = function(attr, oldVal,
12        newVal) {};
12    // Registers custom element
13    document.registerElement('my-element', {
14        prototype: element
15    });
16 })();
17
```

demo1.old-way.js 12:32 LF UTF-8 Babel 0 files

CUSTOM ELEMENTS 2016

The screenshot shows a code editor window with the file 'index.html' open. The title bar indicates the file is at path '~/Documents/custom elements'. The code itself defines a custom element 'app-drawer'.

```
index.html
index.html — ~/Documents/custom elements

6      <title>Custom elements</title>
7      </head>
8
9      <body>
10     <app-drawer></app-drawer>
11     <script>
12       class AppDrawer extends HTMLElement {
13         constructor() {
14           super();
15         }
16       }
17       window.customElements.define('app-drawer', AppDrawer);
18     </script>
19   </body>
20
21 </html>
22
```

The code uses syntax highlighting to distinguish between HTML tags (red), CSS-like classes (orange), and JavaScript code (blue). The cursor is positioned on the closing brace of the constructor function.

SHADOW DOM

```
index.html — ~/Documents/custom elements
index.html

8
9 <body>
10   <app-drawer></app-drawer>
11   <script>
12     class AppDrawer extends HTMLElement {
13       constructor() {
14         super();
15         this.attachShadow({
16           mode: 'open'
17         });
18         this.shadowRoot.innerHTML = `<style></style>`;
19       }
20     }
21     window.customElements.define('app-drawer', AppDrawer);
22   </script>
23 </body>
24
25 </html>
26
```

scoped style



LF UTF-8 HTML 0 files

CUSTOM ELEMENTS LIFE CYCLE

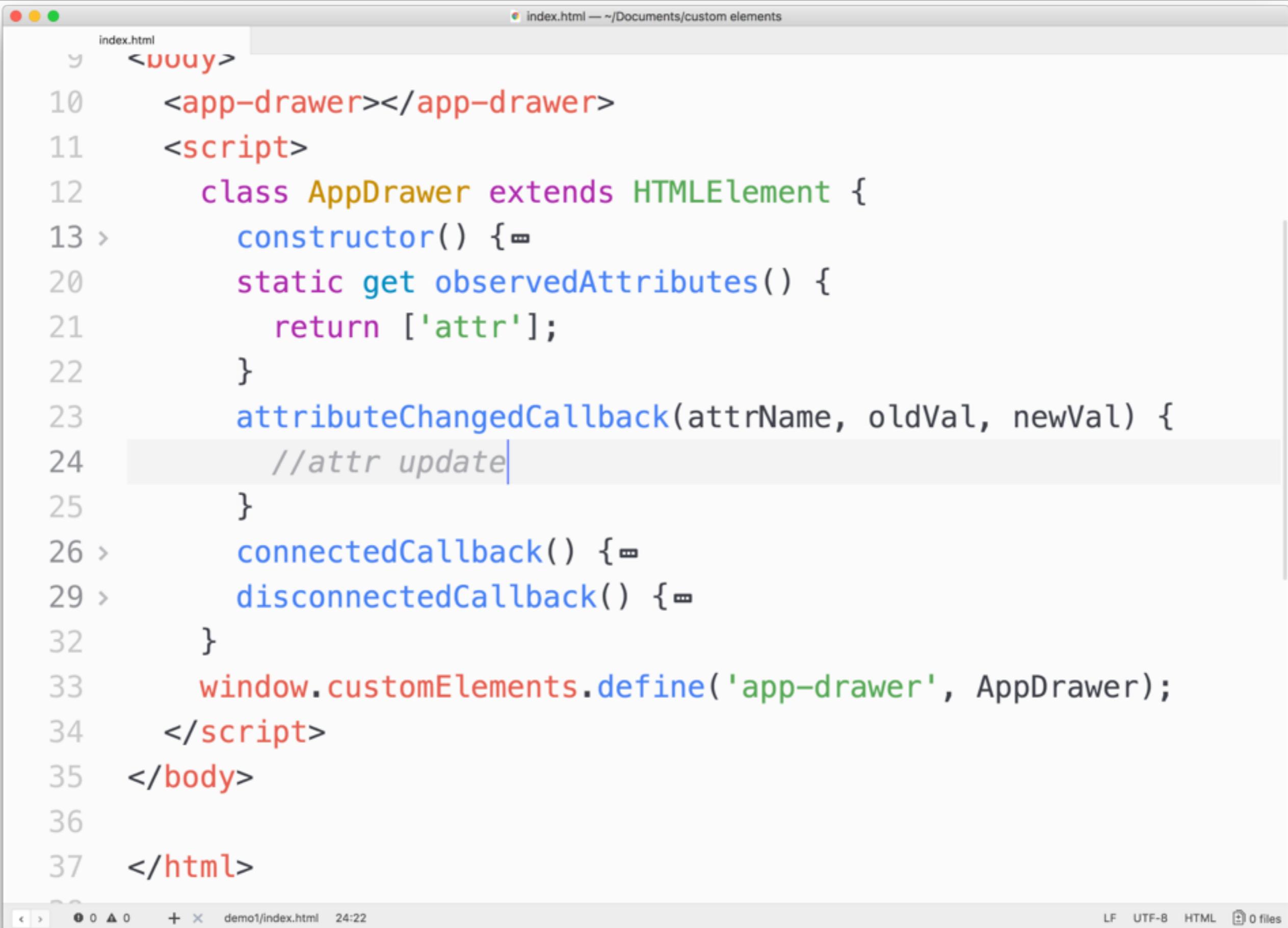
The screenshot shows a code editor window with the file 'index.html' open. The code defines a custom element 'app-drawer' with a constructor and connected/disconnected callbacks. A handwritten blue arrow points from the word 'Life cycle' to the 'connectedCallback' method.

```
index.html — ~/Documents/custom elements
index.html

9 <body>
10   <app-drawer></app-drawer>
11   <script>
12     class AppDrawer extends HTMLElement {
13       constructor() {
14         connectedCallback() {
15           console.log("We are in");
16         }
17         disconnectedCallback() {
18           console.log("We are out");
19         }
20       }
21     }
22   </script>
23 </body>
24
25 </html>
```

Life cycle

CUSTOM ELEMENTS OBSERVING ATTRIBUTES



The screenshot shows a code editor window with the file "index.html" open. The code defines a custom element "app-drawer" that observes the "attr" attribute. The "attributeChangedCallback" function is partially implemented with a placeholder comment "//attr update".

```
index.html
<app-drawer></app-drawer>
<script>
  class AppDrawer extends HTMLElement {
    constructor() {=
      static get observedAttributes() {
        return ['attr'];
      }
      attributeChangedCallback(attrName, oldVal, newVal) {
        //attr update
      }
    connectedCallback() {=
    disconnectedCallback() {=
  }
  window.customElements.define('app-drawer', AppDrawer);
</script>
</body>
</html>
```

index.html — ~/Documents/custom elements

demo1/index.html 24:22

LF UTF-8 HTML 0 files

CUSTOM ELEMENTS TEMPLATES

The screenshot shows a code editor window with a tab labeled "index.html" and a status bar indicating "demo1/index.html 25:42". The code is written in JavaScript and defines a custom element "app-drawer". It includes a template with a style rule and content, and a constructor function that attaches a shadow root and clones the template content. A handwritten note "use template" with a blue arrow points to the line where the template content is cloned.

```
index.html
10  <app-drawer></app-drawer>
11  <template id="template">
12      <style>p { color: orange; }</style>
13      <p>Template content</p>
14  </template>
15  <script>
16      class AppDrawer extends HTMLElement {
17
18          constructor() {
19              super();
20          >      this.attachShadow({=}
21              const t = document.querySelector('#template');
22              const instance = t.content.cloneNode(true);
23              shadowRoot.appendChild(instance);
24
25          }
26
27
28          >      static get observedAttributes() {=}
29          >          attributeChangedCallback(attrName, oldVal, newVal) {=}
30          >              connectedCallback() {=}
```

use template



ЭКСПЕРИМЕНТЫ

CUSTOM ELEMENTS

The screenshot shows a Google Chrome browser window with a dark theme. The title bar includes the standard Apple logo, menu items like File, Edit, View, History, Bookmarks, People, Window, Help, and a tab labeled "New Tab". The address bar has a placeholder "Search or enter a URL". Below the address bar is a bookmarks bar with various links. The main content area displays the "You've gone incognito" screen, which features a circular icon of a person wearing a fedora and glasses. The text "You've gone incognito" is centered below the icon. A descriptive paragraph explains that private browsing allows users to browse privately, with activity saved but downloads and bookmarks preserved. It also notes that activity might still be visible to others. Two columns of text provide details on what information is saved and what might be visible.

You've gone incognito

Now you can browse privately, and other people who use this device won't see your activity. However, downloads and bookmarks will be saved. [Learn more](#)

Chrome won't save the following information:

- Your browsing history
- Cookies and site data
- Information entered in forms

Your activity might still be visible to:

- Websites you visit
- Your employer or school
- Your internet service provider

ЭКСПЕРИМЕНТЫ

REACT

Chrome File Edit View History Bookmarks People Window Help

New Tab

Bookmarks Database Systems... Кодексы и законы math картины 11528 штучок Design DJANGO leasing GAPI JS rent macOS frameworks and tools запроска картридж... красота useful info CI Other Bookmarks



You've gone incognito

Now you can browse privately, and other people who use this device won't see your activity. However, downloads and bookmarks will be saved. [Learn more](#)

Chrome won't save the following information:

- Your browsing history
- Cookies and site data
- Information entered in forms

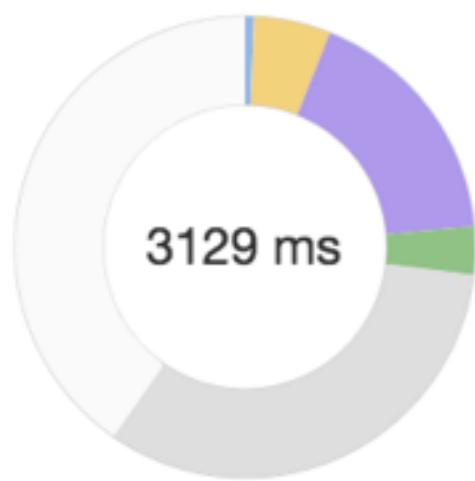
Your activity might still be visible to:

- Websites you visit
- Your employer or school
- Your internet service provider

CUSTOM ELEMENTS VS REACT ЗАГРУЗКА

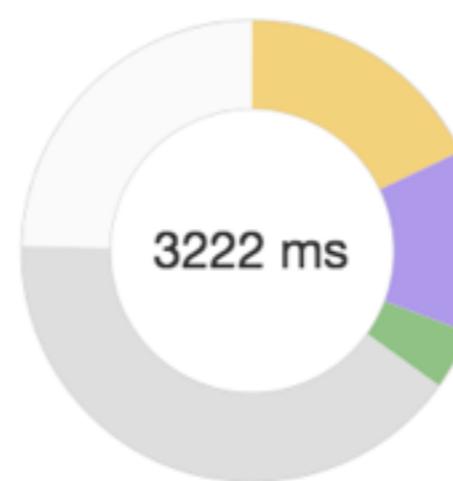
Range: 0 – 3.13 s

CUSTOM ELEMENTS

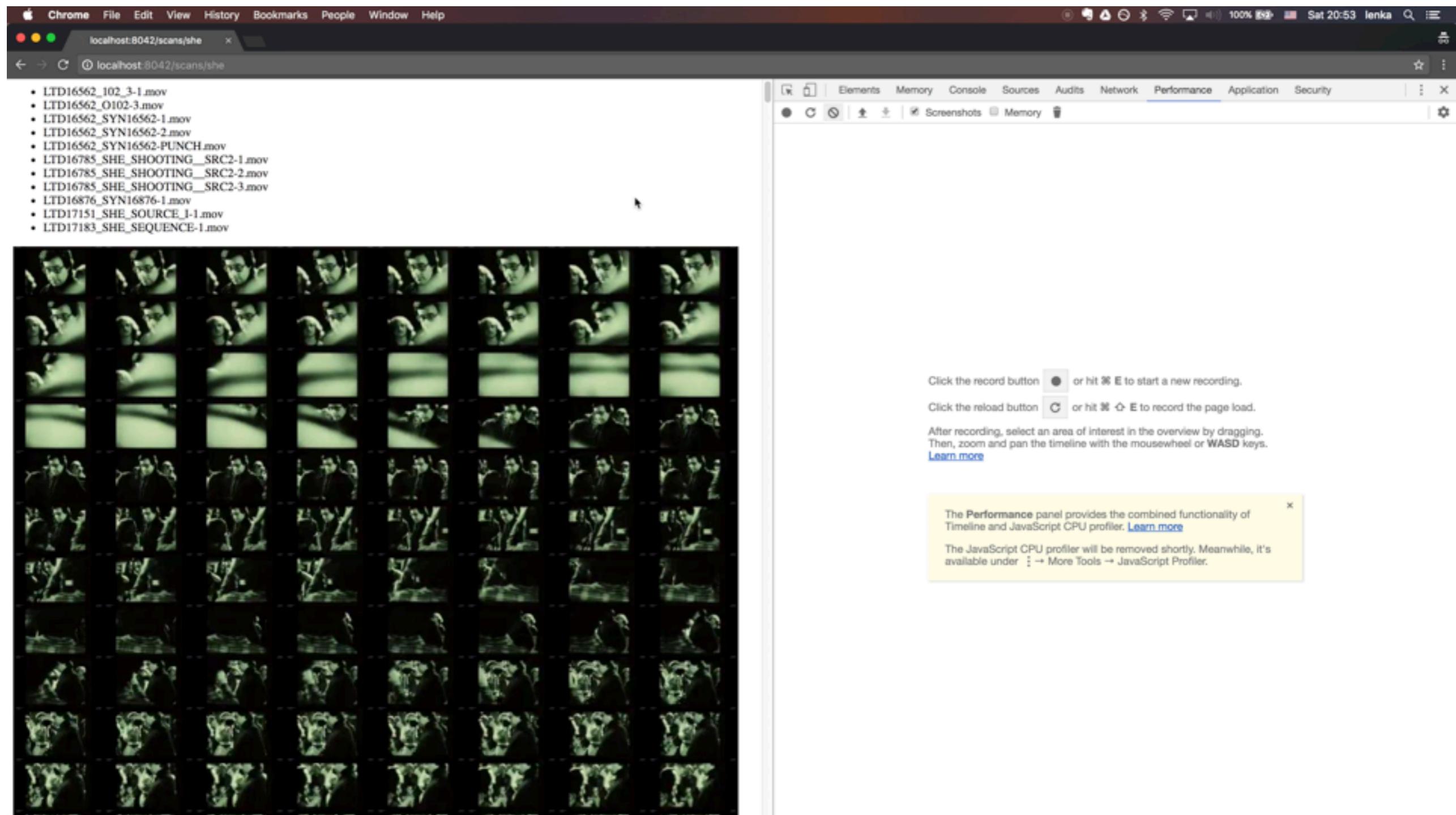


Range: 0 – 3.22 s

REACT



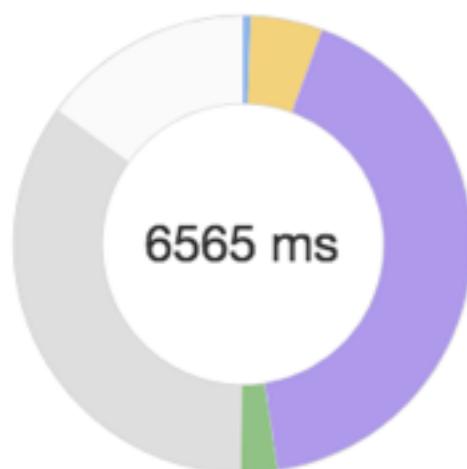
ЭКСПЕРИМЕНТЫ



CUSTOM ELEMENTS VS REACT ПЕРЕРИСОВКА

Range: 1.88 s – 8.45 s

CUSTOM ELEMENTS



37.4 ms	■ Loading
332.3 ms	■ Scripting
2754.0 ms	■ Rendering
168.3 ms	■ Painting
2290.0 ms	■ Other
983.2 ms	■ Idle

■ Event

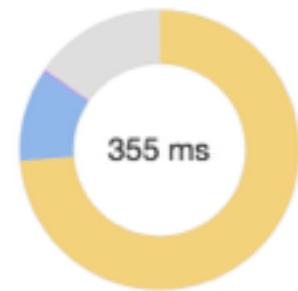
Warning Handler took 354.95 ms

Total Time 354.95 ms

Self Time 0.30 ms

Type click

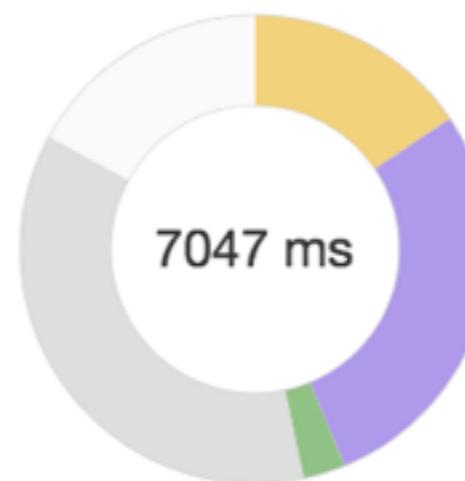
Aggregated Time



0.3 ms	■ Scripting (self)
261.6 ms	■ Scripting (children)
37.4 ms	■ Loading
1.5 ms	■ Rendering
54.2 ms	■ Other

Range: 2.78 s – 9.83 s

REACT



1096.8 ms	■ Scripting
1989.2 ms	■ Rendering
204.0 ms	■ Painting
2559.4 ms	■ Other
1197.4 ms	■ Idle

■ Event

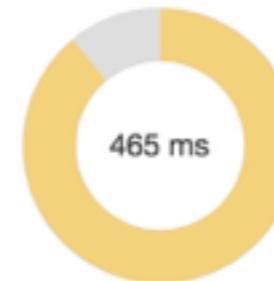
Warning Handler took 464.51 ms

Total Time 464.51 ms

Self Time 0.21 ms

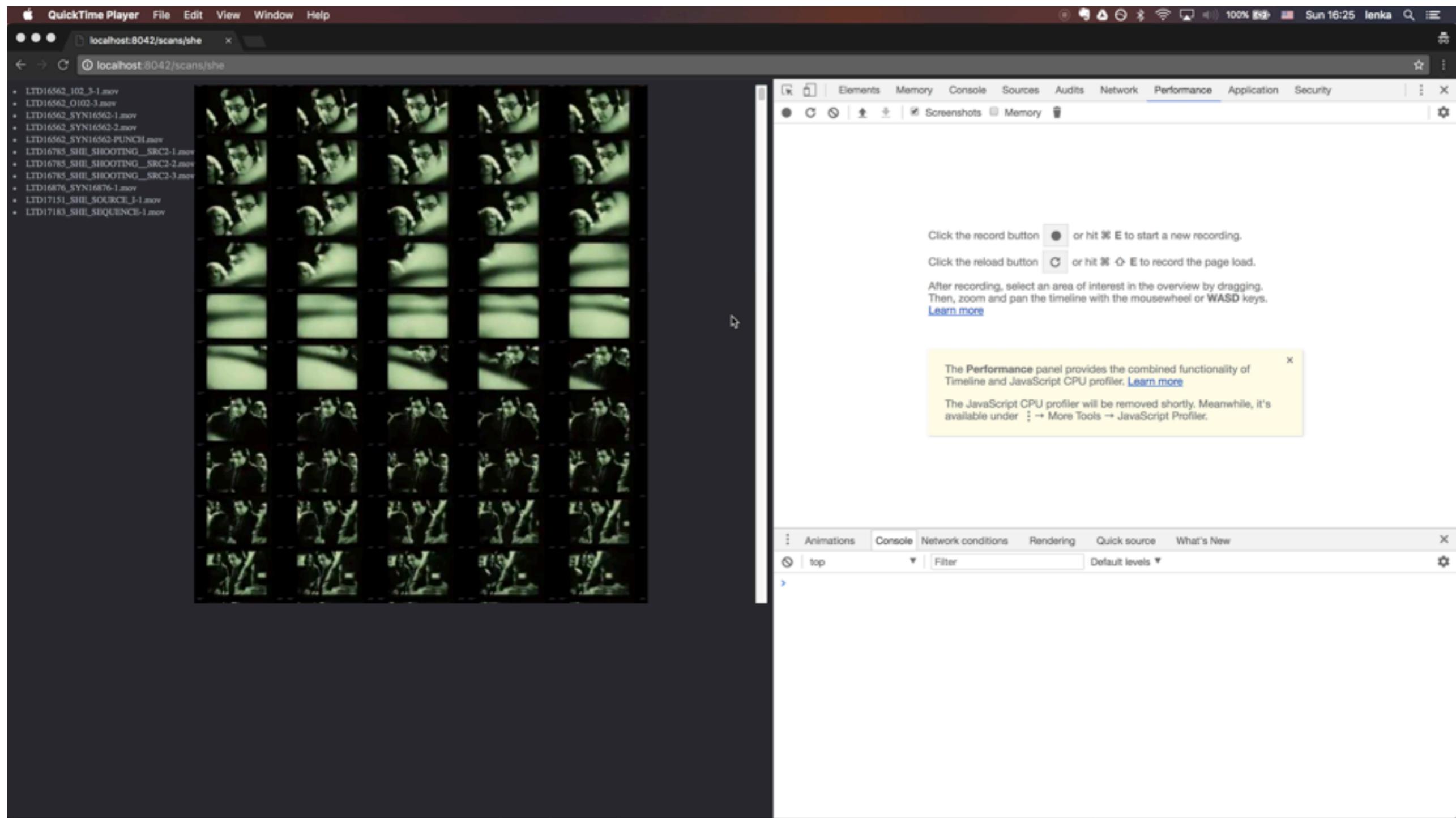
Type click

Aggregated Time



0.2 ms	■ Scripting (self)
413.5 ms	■ Scripting (children)
50.9 ms	■ Other

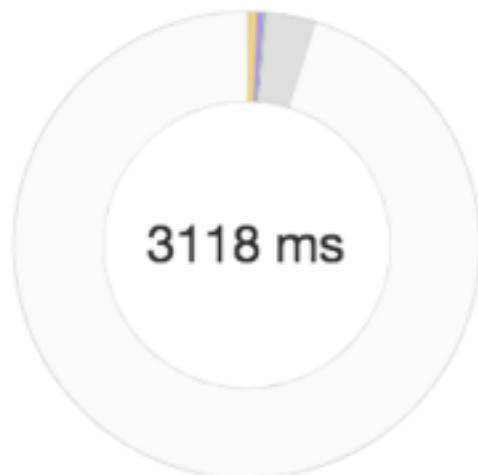
ЭКСПЕРИМЕНТЫ



CUSTOM ELEMENTS VS REACT

Range: 0 – 3.12 s

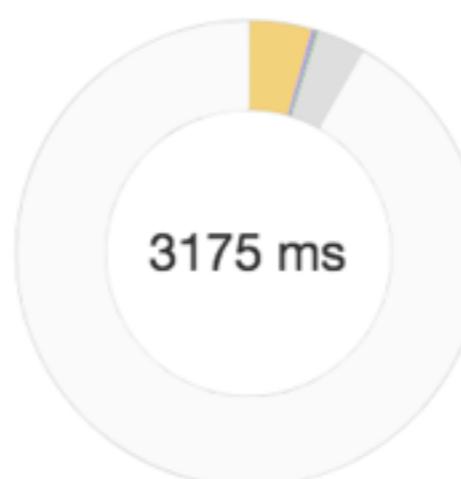
CUSTOM ELEMENTS



2.1 ms	■ Loading
19.0 ms	■ Scripting
14.7 ms	■ Rendering
5.8 ms	■ Painting
109.2 ms	■ Other
2967.2 ms	□ Idle

Range: 0 – 3.17 s

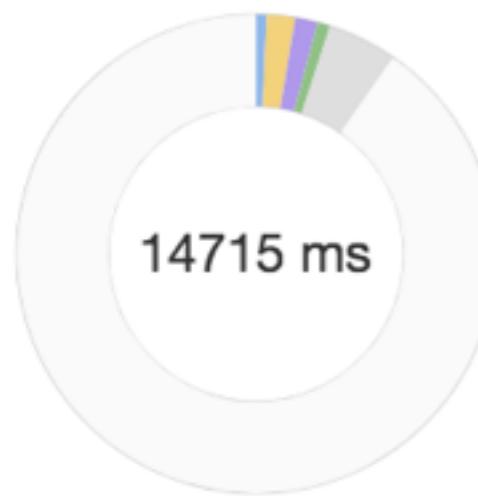
REACT



2.2 ms	■ Loading
137.7 ms	■ Scripting
9.0 ms	■ Rendering
5.6 ms	■ Painting
107.7 ms	■ Other
2912.3 ms	□ Idle

Range: 0 – 14.72 s

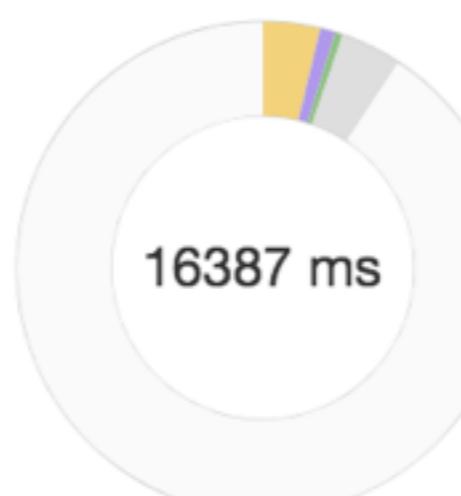
CUSTOM ELEMENTS



106.0 ms	■ Loading
279.9 ms	■ Scripting
224.0 ms	■ Rendering
126.0 ms	■ Painting
690.8 ms	■ Other
13288.3 ms	□ Idle

Range: 0 – 16.39 s

REACT



615.6 ms	■ Scripting
164.0 ms	■ Rendering
81.5 ms	■ Painting
662.3 ms	■ Other
14863.4 ms	□ Idle

УДОБСТВО РАЗРАБОТКИ

УДОБСТВО РАЗРАБОТКИ

```
video-files.js — ~/vg-brain
video-files.js

33
34  export default class VideoFiles extends fileList {
35 >    constructor (files) {=
36
37
38
39
40    render () {
41 >      if (!this._files) {=
42        const fragment = document.createDocumentFragment();
43        this._files.forEach(file => {
44          let fileItem = new VideoItem(file);
45          //...
46          fragment.appendChild(fileItem);
47        });
48      }
49      this.fileList.appendChild(fragment);
50    }
51  }
52
53
54  window.customElements.define("video-files", VideoFiles);
55
```

OOP style

УДОБСТВО РАЗРАБОТКИ

The screenshot shows a code editor window with the file 'video-files.js' open. The code is written in JavaScript and includes a CSS template string. Handwritten text 'CSS with CSS' is written in blue ink over the code, with a blue arrow pointing from the text towards the opening tag of the style block.

```
video-files.js — ~/vg-brain
video-files.js

5 const WIDTH = 120.0;
6 const HEIGHT = 68.0;
7 const template = document.createElement('template');
8 template.innerHTML =
9 <style>
10 >   :host{-
13 >     video-item{-
18 >       video-item:hover{-
21 >         video-item[selected=true]{-
25 >           video-item img {-
31
32 </style>`;
33
34 > export default class VideoFiles extends fileList {-
54
55 window.customElements.define("video-files", VideoFiles);
56
```

Handwritten text: CSS with CSS

File status: LF UTF-8 Babel scans-test 3 2 files

ROB DODSON



CUSTOM EVENTS

```
file-list.js — ~/vg-brain
12  export default class FileList extends HTMLElement {
13 >    constructor(files) {=
22
23 >    set files(files) {=
27
28     selectFile(item, file) {
29 >        if (this._selectedFileItem) {=
34         this.dispatchEvent(new CustomEvent('file-select',
35             {bubbles: true, composed: true, detail: file}));
36
37 >    render() {=
49
50    }
51    window.customElements.define('file-list', FileList);
52
```

ЗАКЛЮЧЕНИЕ

Модульность



Производительность

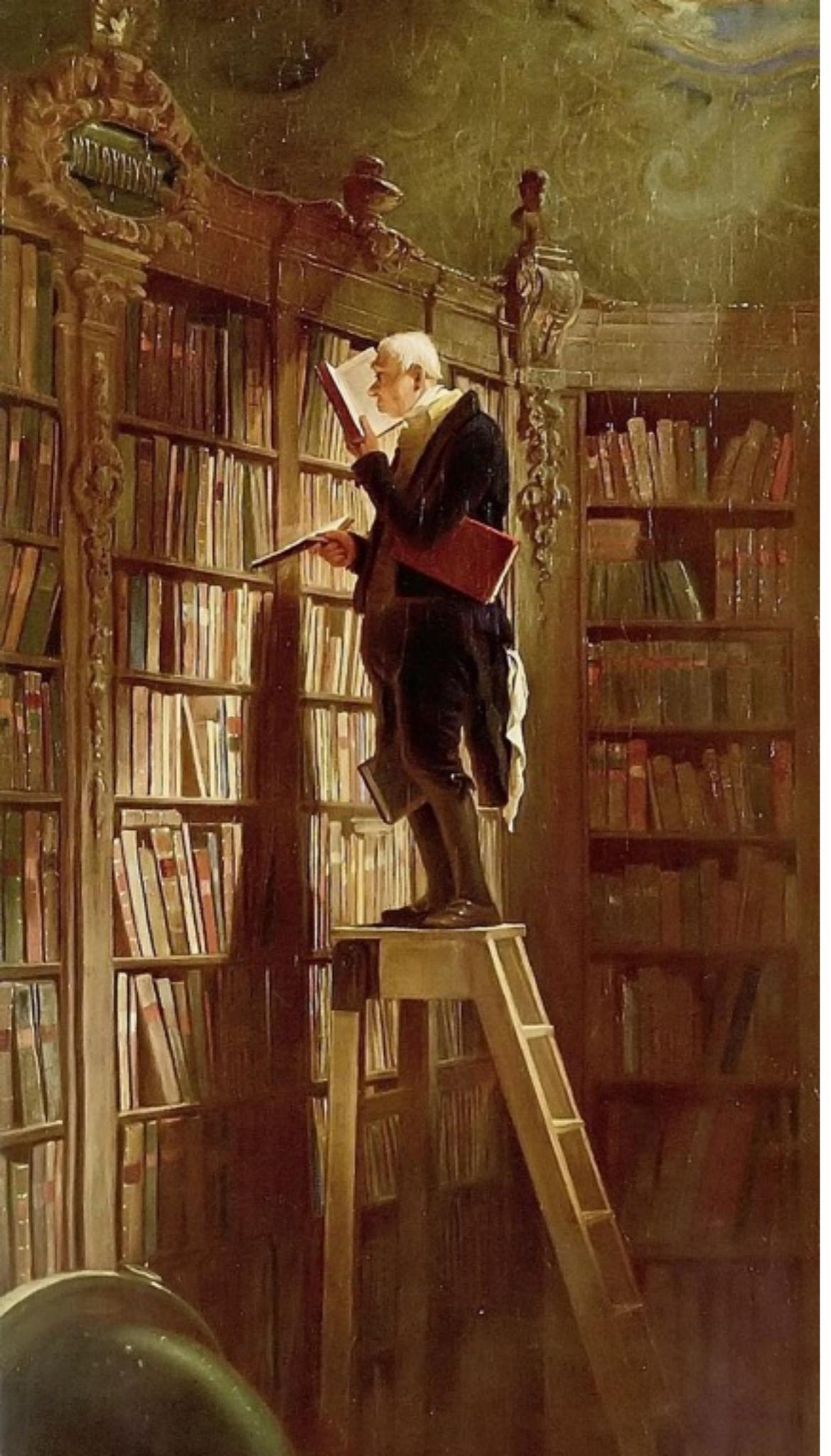


Удобство разработки



Кросбраузерность





источники

ИСТОЧНИКИ

WEBCOMPONENTS.ORG

← → C Secure | https://www.webcomponents.org ⋮

Search custom elements IMAGE LAYOUT NOTIFICATION ROUTING MEDIA

Featured elements [Browse elements](#)

canvas-datagrid

Canvas based data grid web component. Capable of displaying millions of

 TonyGermaneri

simple-dropdown

Featherweight, style-agnostic dropdown UI component

 SimpleElements

paper-dynamic-forms

This component gives an easy and comfortable way to make dynamics

 alfonsorios96

Featured collections [Browse collections](#)

cr-elements

4 items

The Clash Royale elements are a set of UI components designed to implement Clash

 Dabolus

granite-elements

19 items

A collection of web-components made on the granite coast of Brittany

 LostInBrittany

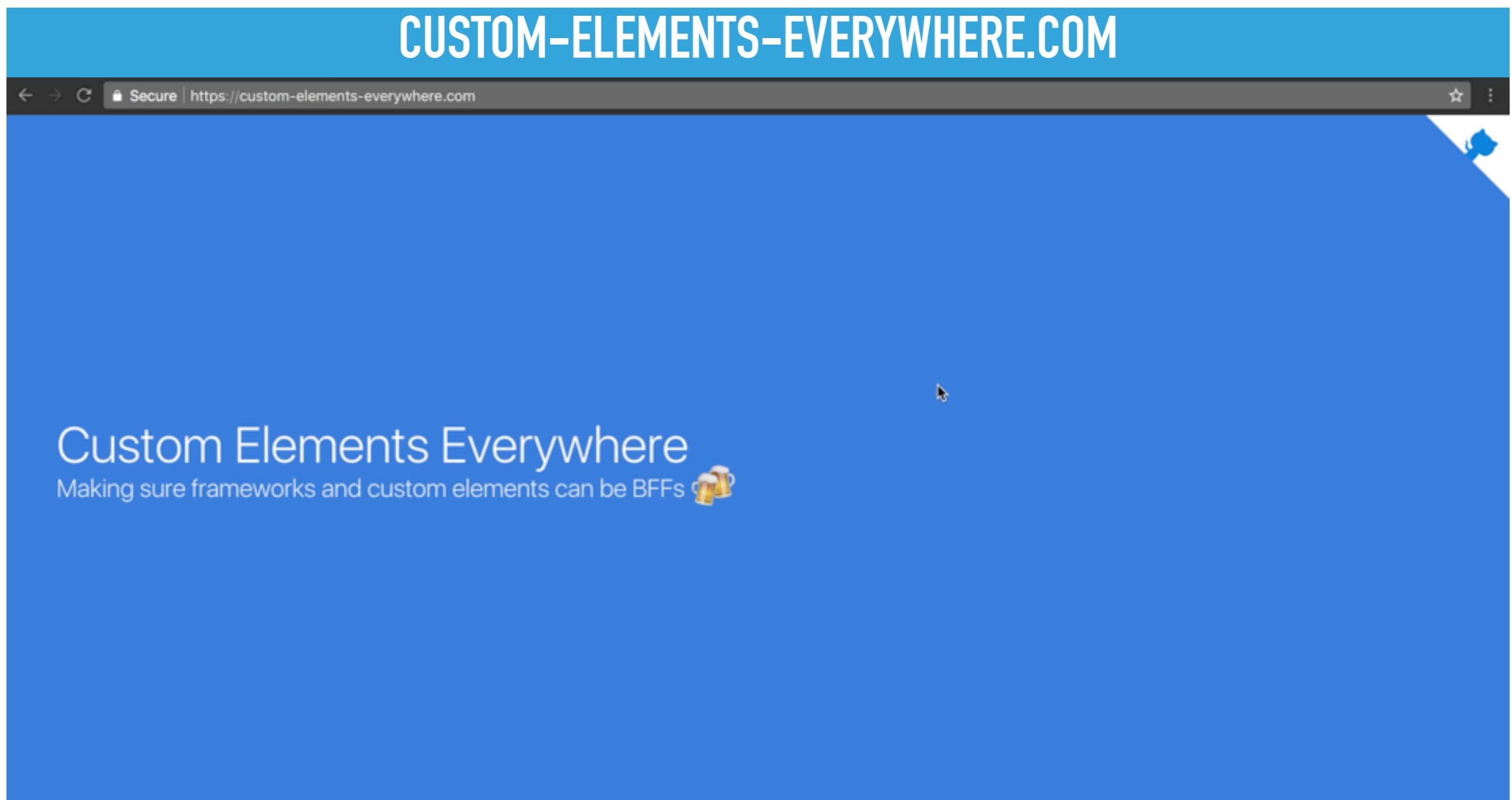
fs-elements

21 items

Collection of FamilySearch Polymer elements.

 fs-webcomponents

ИСТОЧНИКИ



What's this?

Custom Elements are the lynchpin in the Web Components specifications. They give developers the ability to define

ИСТОЧНИКИ

The screenshot shows a web browser window with a dark theme. The address bar displays "Secure | https://robododson.me/author/rob/". The main content area features a circular profile picture of a man with glasses and a mustache, identified as Rob Dodson. Below the picture is the name "Rob Dodson" in a large, bold, dark font. Underneath the name is a small icon of a bar chart followed by the text "104 posts". A horizontal line with a central circle follows. The first post title, "The future of accessibility for custom elements", is displayed in a large, bold, dark font. Below it is a short description: "When users of assistive technology, like a screen reader, navigate a web page, it's vitally important that the semantic meaning of the various controls is »". Underneath the post details is another horizontal line with a central circle. The second post title, "How to use Polymer with Webpack", is displayed in a large, bold, dark font. Below it is a short description: "Introduction Over the last year I've had a number of discussions with mid to large sized companies who are interested in creating common UI libraries that »". Both posts include a small circular profile picture of Rob Dodson, the author's name, and the date of publication.

Rob Dodson - Page 1 - Rob Do X

Secure | https://robododson.me/author/rob/

MENU

104 posts

The future of accessibility for custom elements

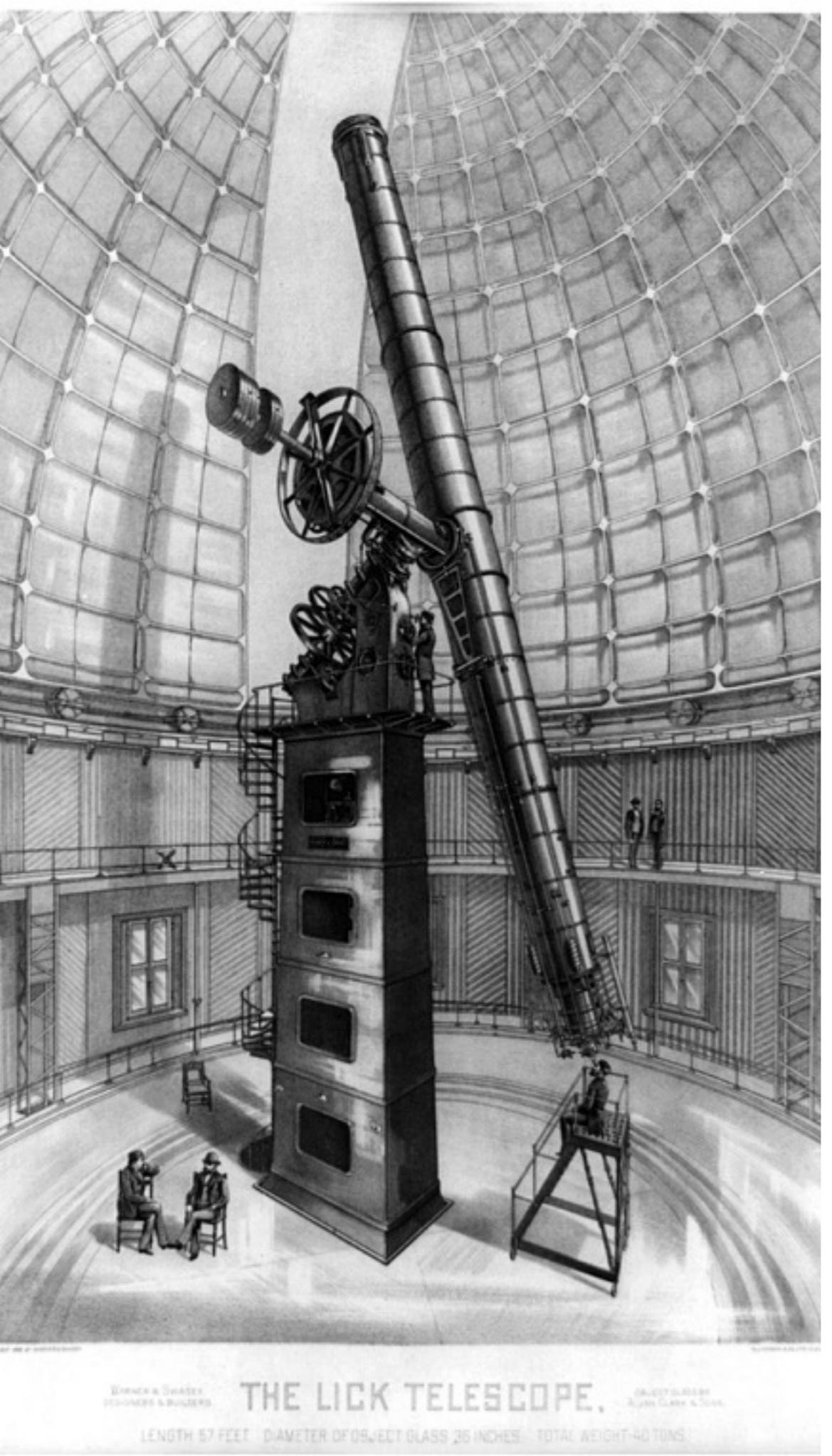
When users of assistive technology, like a screen reader, navigate a web page, it's vitally important that the semantic meaning of the various controls is »

Rob Dodson | 02 OCTOBER 2017

How to use Polymer with Webpack

Introduction Over the last year I've had a number of discussions with mid to large sized companies who are interested in creating common UI libraries that »

Rob Dodson | 17 JULY 2017



СВЕТЛОЕ БУДУЩЕЕ?

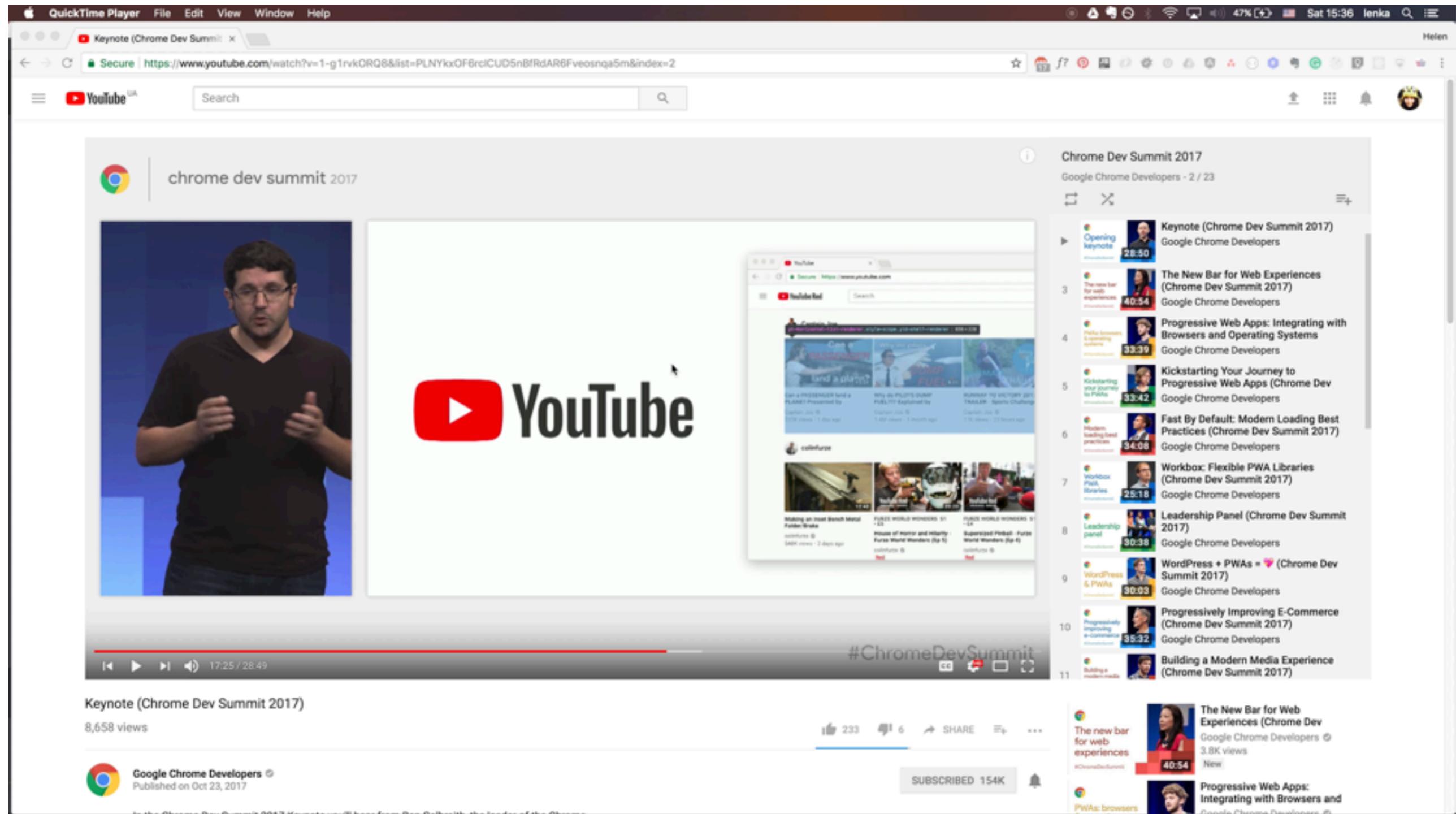
BARNARD & SWARTZ
DESIGNERS & BUILDERS

THE LICK TELESCOPE.

MADE BY
ALLEN CARL & CO.

LENGTH 57 FEET - DIAMETER OF OBJECT GLASS 36 INCHES - TOTAL WEIGHT 40 TONS

СВЕТЛОЕ БУДУЩЕЕ?



СВЕТЛОЕ БУДУЩЕЕ?

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

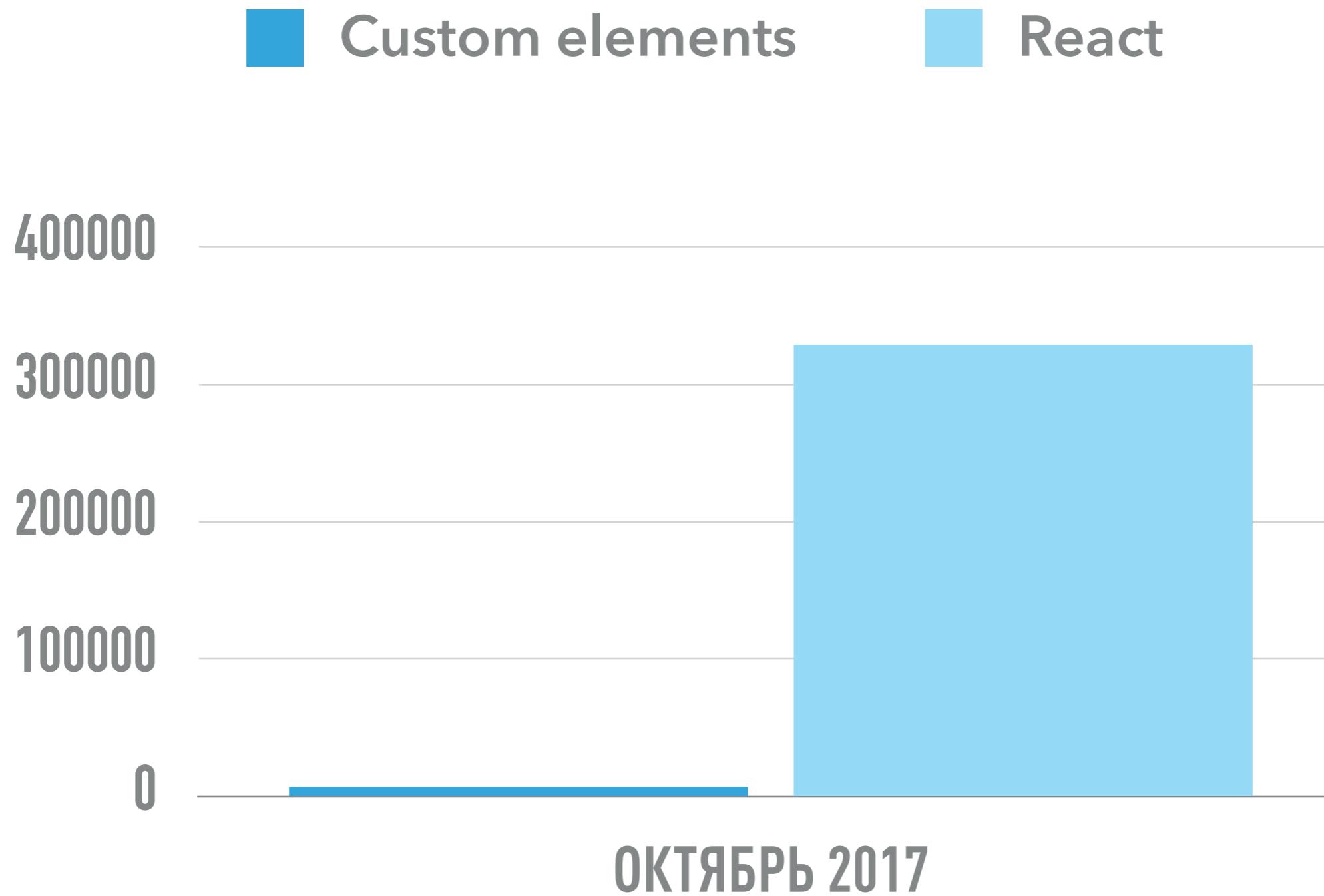
STABLE

POLYFILL
ON HOLD

POLYFILL
ON HOLD

POLYFILL
CONSIDERING

СВЕТЛОЕ БУДУЩЕЕ?



2006

jQuery

2009



2010



BACKBONE.JS

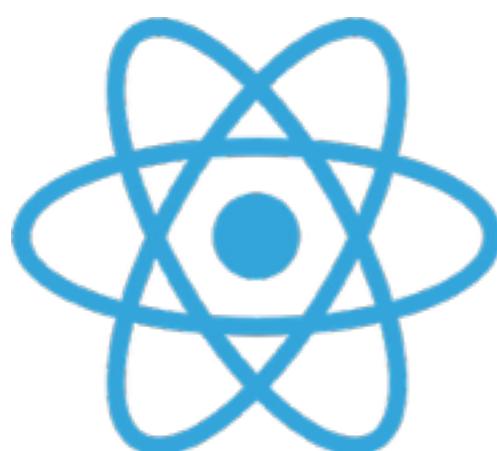
2011

ember

2012



2013



2014



2015



2016



SWITZERLAND



2017