Web Components—What’s the Catch?

TJ VanToll | @tjvantoll
UI libraries are seen as the ideal use case for web components
Proof-of-concept rewrite of a few jQuery UI widgets to use web components

https://github.com/tjvantoll/ui-web-components
Web components’ public image

• “[T]he Web Components revolution”

• “Web components are a game changer”

• “Web Components Are The Future Of Web Development”
  – [http://techcrunch.com/2013/05/19/google-believes-web-components-are-the-future-of-web-development/](http://techcrunch.com/2013/05/19/google-believes-web-components-are-the-future-of-web-development/)
Web components’ public image

- “A Tectonic Shift for Web Development”
  - https://developers.google.com/events/io/2013/sessions/318907648

- “Join the Web Components revolution”

- “Web Components usher in a new era of web development”
  - https://www.polymer-project.org/
Web components’ public image

- “Web Components - A Quantum Leap in Web Development”
  - http://lanyrd.com/2014/qconsf/sddqfk/
- “The Dawn of the Reusable Web”
- “Web Components are ushering in a HTML renaissance”
The catch

• Polyfilling shadow DOM
• Resolving HTML import dependencies
• Changing form elements’ UI
• Browser support
The catch

• Polyfilling shadow DOM
• Resolving HTML import dependencies
• Changing form elements’ UI
• Browser support
Shadow DOM
(native behavior in Chrome)
Shadow DOM (polyfilled behavior in Safari)

```html
<html class="no-js">
    <!-<! [endif]-->
    <head>...</head>
<body>
    <div class="hero-unit">
        < polymer-greeting>
            <h1>'Allo, 'Allo!</h1>
            <span>Update text to change the greeting.</span>
            <input type="text">
        </polymer-greeting>
    </div>
</body>
```

```
> document.querySelectorAll( "h1" ).length
< 0
```
Shimming DOM APIs

```javascript
setQuerySelectorAll: function(selector) {
    var shimmed = shimSelector(selector);
    var deep = shimmed !== selector;
    selector = shimmed;

    var result = new NodeList();

    result.length = querySelectorAllFiltered.call(this,
        matchesSelector,
        0,
        result,
        selector,
        deep);

    return result;
}
```
Shim all the things!

https://github.com/Polymer/ShadowDOM/tree/master/src/wrappers
Polyfilling CSS selectors

• The shadow DOM specification introduces a lot of new CSS things.
  – ::shadow, ::content, /deep/, etc

• Browsers discard CSS selectors, properties, and rules they don’t understand.

• For polyfilling your only recourse is to run text searches on CSS files.
```javascript
var selectorRe = /([^]*)\([^\s\S]*\))/gim,
cssCommentRe = /\s*\([^\s\S]*\)+([^\s\S]*\)+\s+/gim,
  // TODO(sorvell): remove either content or comment
cssCommentNextSelectorRe = /\s*\([^\s\S]*\)+([^\s\S]*\)+\s+/gim,
cssContentNextSelectorRe = /polyfill-next-selector\([^\s\S]*\)+content:\[\s*\]?'.'?'.'?'['\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]'\]';\]
The intention here is to support only the styling features which can be relatively simply implemented. The goal is to allow users to avoid the most obvious pitfalls and do so without compromising performance significantly. For ShadowDOM styling that's not covered here, a set of best practices can be provided that should allow users to accomplish more complex styling.
Why this matters?

• It’s hard to tell what is polyfilled and what isn’t

• Complexity adversely affects file size and performance
  – The latest version of the shadow DOM polyfill is 69KB minified, 19KB minified and gzipped.

• Chrome is the only browser shipping shadow DOM.
Polymer

• Polymer 0.8 no longer requires the shadow DOM polyfill

Highlights

- Dramatically faster startup time and runtime performance than 0.5, even in Chrome where web components are natively supported.
- Significantly smaller payload than 0.5.
- Completely refactored internally to be clearly layered and much less complex.
- Brand new data-binding system that is simpler, faster, offers tighter control, and is easier to debug.
- Brand new, lightweight shadow DOM shim called shady DOM, that lets you avoid the complexity, size, performance penalty, and invasiveness of the shadow DOM polyfill.
- Upper bound and lower bound scoped styling, even without native shadow DOM: scoped styles don’t bleed out, and children in their own roots are protected from descendant selectors in a shadow root.

https://www.polymer-project.org/0.8/
The catch

• Polyfilling shadow DOM
• Resolving HTML import dependencies
• Changing form elements’ UI
• Browser support
HTML Imports

<link rel="import" href="/path/to/ui-progressbar.html">
<ui-progressbar value="10" max="100"></ui-progressbar>

Awesomely concise and easy to use syntax, but...
HTML Imports

• How to depend on a third-party resource—e.g. Bootstrap, jQuery, Moment, Polymer, etc?

• Deduping
  – If two components request a resource from the same URL, supporting browsers are smart enough to suppress the second request.
What URL to use?

- Only avoids duplicate requests if path is EXACTLY the same (i.e. same protocol, host, version, file name)

- Enforces a folder structure on users of your component. Also assumes component authors always use the same file name (i.e. moment.js and not moment.min.js)
Firefox’s decision

• “Mozilla will not ship an implementation of HTML Imports. We expect that once JavaScript modules — a feature derived from JavaScript libraries written by the developer community — is shipped, the way we look at this problem will have changed.”

https://hacks.mozilla.org/2014/12/mozilla-and-web-components/
The catch

- Polyfilling shadow DOM
- Resolving HTML import dependencies
- Changing form elements’ UI
- Browser support
Full video game engine with 3D rendering and real-world physics in a browser? Yes. Ability to style <select> dropdowns in a browser? No.
Web components are seen as a panacea for these type of issues.

Nicholas C. Zakas @slicknet · Jul 2
Full video game engine with 3D rendering and real-world physics in a browser? Yes. Ability to style <select> dropdowns in a browser? No.

Oleg Aleynik @oaleynik · Jul 2
@slicknet @Web_Components to the rescue! youtube.com/watch?v=liK-7Q...

Jakob @jakobo
@oaleynik @slicknet Web Components are going to let us make the standard UI tags we wish we had. The good ones will be accessible too :D
Good news: You can write custom elements that extend native form elements
Bad news: Custom elements don’t give you any special control over the UI of native form elements.
mm/dd/yyyy
• Chrome 31’s shadow DOM implementation lets you replace the shadow root of `<input>` elements, which lets you alter the default UI.

```html
<input type="date">
```

```css
input {
  height: 2em;
  width: 15em;
}
```

```javascript
var dateRoot = document.querySelector("input").createShadowRoot();

$("input").datepicker({
  dateFormat: "yy-mm-dd",
  onSelect: function(dateText) {
    dateRoot.innerHTML = dateText;
  }
});
```
• **Bad news**: When you remove an input’s shadow root that element loses its native functionality:
  – You can’t type in it
  – There’s no HTML5 form validation
  – It’s basically not an `<input>` anymore.

• **Bottom line**: Web components don’t currently offer a solution to styling form elements.
Form elements are a common reason people use UI libraries

- jQuery UI
  - Autocomplete
  - Datepicker
  - Selectmenu
  - Slider
  - Spinner

- Kendo UI
  - AutoComplete
  - ColorPicker
  - ComboBox
  - DatePicker
  - DateTimePicker
  - DropDownList
  - Editor
  - MultiSelect
  - NumericTextBox
  - Slider
  - TimePicker
  - Upload
The catch

- Polyfilling shadow DOM
- Resolving HTML import dependencies
- Changing form elements’ UI
- Browser support
Browser support

• One of the biggest things holding back web components is lack of browser support.
  – Related: water is wet; the sun is hot
• jQuery UI / Kendo UI have to be very cognizant of file size and almost never use polyfills other than those in jQuery Core.
• The web components polyfills don’t support a lot of the browsers jQuery UI / Kendo UI support today (IE 8–10, Android 2.3–4.4)\(^1\).

1: [https://github.com/webcomponents/webcomponentsjs/#browser-support](https://github.com/webcomponents/webcomponentsjs/#browser-support)
My advice

• Think of web components as 4 separate technologies that may be able to help your apps.
  – Shadow DOM
  – HTML imports
  – `<template>`
  – Custom elements
Shadow DOM

- Complex, big, and slow polyfill
- Chrome has the only implementation
  - Firefox working on supporting but working through a number of issues¹
- Provides more value to frameworks than individual apps
- My recommendation: Hold off

HTML Imports

• Chrome has the only stable implementation
  – Firefox is holding off¹

• No good solution to manage third-party dependencies

• My recommendation: Hold off

¹: https://hacks.mozilla.org/2014/12/mozilla-and-web-components/
• Chrome, Firefox, and Safari all have implementations
• A way to add inert DOM elements to your document
  – Most common use case is a replacement for `<script type="text/html">`
• Not something that’s going to revolutionize your apps
• My recommendation: **Hold off** until browser support is ubiquitous
Custom elements

• Chrome has the only implementation
• Relatively easy to polyfill
  – Polyfills have much better browser support
• Provide a lot of value (creating custom HTML elements) with relatively little downside
• My recommendation: *Use!*
Custom elements polyfills

- [https://github.com/webcomponents/webcomponentsjs/](https://github.com/webcomponents/webcomponentsjs/)
  - Evergreen browsers, IE9+, Android 4+

  - Evergreen browsers, IE8+, iOS 5+, Android 2.2+
  - 2.5K!
Custom elements API improvements

```html
<input id="datepicker">
<script>
  \$("#datepicker").datepicker();
</script>

<ui-datepicker></ui-datepicker>

<input id="datepicker">
<script>
  \$("#datepicker").kendoDatePicker();
</script>

<kendo-datepicker></kendo-datepicker>
Why I’m personally excited for custom elements

• One of the more common requests we get at jQuery UI and Kendo UI is for integration with or compatibility with {{ framework }}.
I suggest you ...

Integrate with Ember.js

It would be great if there was a wrapper library for Ember.js similar to the kendo-labs libraries for Angular and Knockout.

Jacob Jewell shared this idea · September 12, 2013 · Flag idea as inappropriate...
I suggest you ...

← Telerik Kendo UI Feedback

Make an example/blog post about Kendo UI Web + Facebook React

React uses a declarative paradigm that makes it easier to reason about a JavaScript application. React computes the minimal set of changes necessary to keep your DOM up-to-date.

Kendo UI seems not to have such an update feature. But since React works with most libraries and frameworks, please create an example app with React or make a blog post about how to use React together with Kendo UI Web.

Anonymous shared this idea · July 17, 2013 · Flag idea as inappropriate...
Complete Knockout support

Let Kendo work with KnockoutJS.

Anders shared this idea · February 24, 2012 · Flag idea as inappropriate...
Kendo UI provides official Angular directives for both Kendo UI Core and Professional

• **Kendo UI Core**
  – 30+ widgets
  – Free to use and open source

• **Kendo UI Professional**
  – ~10 widgets
    • Grid, Charts, Schedulers, Editors
  – Commercial product
AngularUI

The companion suite(s) to the AngularJS framework.
React-widgets offers a set UI widgets, built from scratch with React. The suite is based on the excellent work done by Kendo UI Core, and jQuery UI, but built as true components, and not library wrappers. By building each widget entirely in React, it can leverage all of the benefits of the React ecosystem and philosophy.

A big thanks to both of these libraries for solving most of the difficult problems already, and providing an excellent reference for what works, and what does not, in ui inputs.
It’s silly to reinvent the wheel every time a new framework comes out.
Wrapping up

• Ignore the hype; evaluate web components for yourself.
• Try each of the web components technologies individually.
• Start with custom elements.
Resources

• Guide to using custom elements

• Simple custom element example—<clock-face>
  – https://github.com/tjvantoll/clock-face

• GitHub’s extension to the <time> element
  – https://github.com/github/time-elements

• Angelina Fabbro: Web Components—Drunk on the Panacea (more detail than I’ve been able to get into)
  – https://vimeo.com/110972839
Thanks!

TJ VanToll

http://tjvantoll.com
http://twitter.com/tjvantoll