

# Web Components— What's the Catch?

TJ VanToll | @tjvantoll



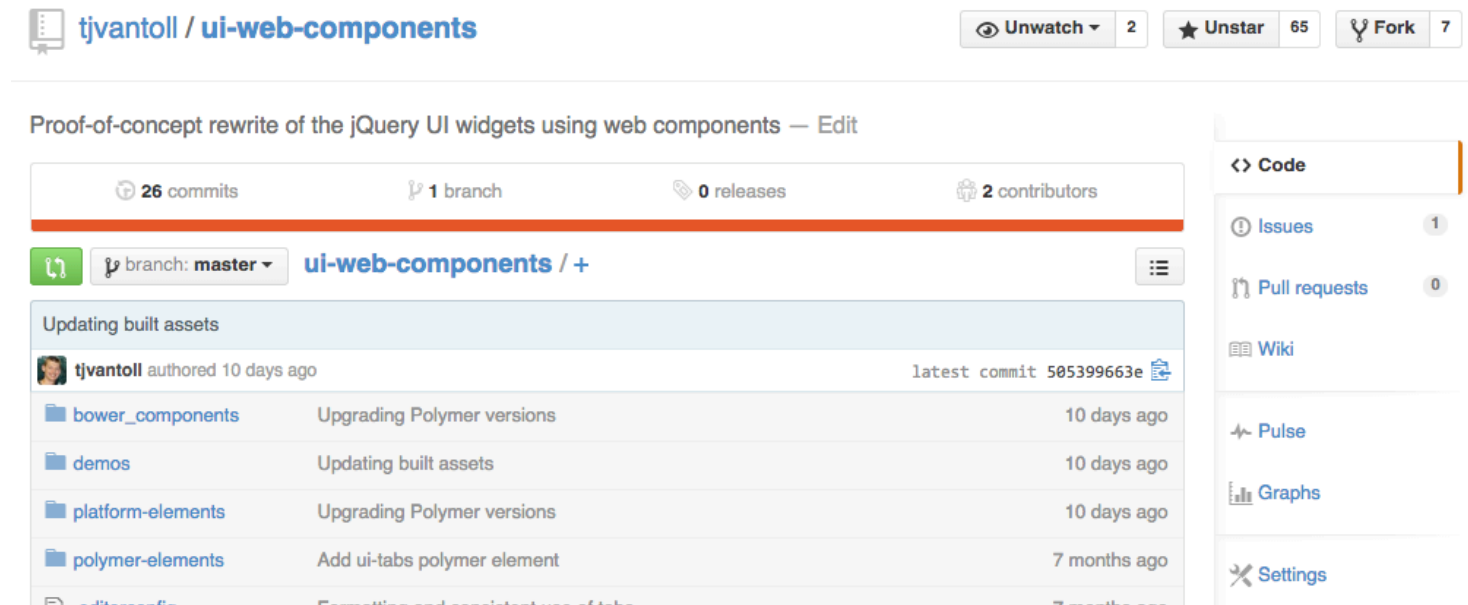
Kendo UI



jQuery UI

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



The screenshot shows the GitHub repository page for 'tjvantoll / ui-web-components'. At the top, it displays the repository name, a 'Unwatch' button with a dropdown arrow, a count of '2' watchers, an 'Unstar' button with a dropdown arrow, a count of '65' stars, and a 'Fork' button with a dropdown arrow and a count of '7' forks. Below this, the repository description reads 'Proof-of-concept rewrite of the jQuery UI widgets using web components — Edit'. A summary bar shows '26 commits', '1 branch', '0 releases', and '2 contributors'. The main content area shows the 'master' branch selected, with a list of commits. The most recent commit is by 'tjvantoll' from 10 days ago, with the message 'Updating built assets' and the latest commit hash '50539963e'. Below this, a table lists several folders and their commit messages and dates:

Folder	Commit Message	Time Ago
bower_components	Upgrading Polymer versions	10 days ago
demos	Updating built assets	10 days ago
platform-elements	Upgrading Polymer versions	10 days ago
polymer-elements	Add ui-tabs polymer element	7 months ago

On the right side, a sidebar contains navigation links: 'Code', 'Issues' (1), 'Pull requests' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'.

<https://github.com/tjvantoll/ui-web-components>

# Web components' public image

- “[T]he Web Components revolution”
  - <http://webcomponents.org/presentations/polymer-and-the-web-components-revolution-at-io/>
- “Web components are a game changer”
  - <http://webcomponents.org/presentations/polymer-and-web-components-change-everything-you-know-about-web-development-at-io/>
- “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”
  - <http://www.ibm.com/developerworks/library/wa-polymer/>
- “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/qconf/sddqfk/>
- “The Dawn of the Reusable Web”
  - <http://www.codemash.org/session/the-dawn-of-the-reusable-web-diving-into-web-components/>
- “Web Components are ushering in a HTML renaissance”
  - <http://addyosmani.com/blog/video-componentize-the-web-talk-from-lxjs/>

# 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)

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

html.no-js body div.hero-unit polymer-greeting #shadow-root **h1**


Console Search Emulation Rendering

<top frame>

```
> document.querySelectorAll( "h1" ).length
0
```

# Shadow DOM (polyfilled behavior in Safari)

```
▼ <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>
```

 Console

```
> document.querySelectorAll( "h1" ).length
```

```
< 0
```

# Shimming DOM APIs

```
193     querySelectorAll: function(selector) {
194         var shimmed = shimSelector(selector);
195         var deep = shimmed !== selector;
196         selector = shimmed;
197
198         var result = new NodeList();
199
200         result.length = querySelectorAllFiltered.call(this,
201             matchesSelector,
202             0,
203             result,
204             selector,
205             deep);
206
207         return result;
208     }
209 };
```

# Shim all the things!

branch: master webcomponentsjs / src / ShadowDOM / wrappers / +

Revert "Add an activeElement getter for #110"

garlcnation authored 20 days ago latest commit 25e7485bdb

..		
CanvasRenderingContext2D.js	Consistent license headers	4 months ago
CharacterData.js	Patch #225 with a correct setter for nodeValue	22 days ago
DOMTokenList.js	don't fail if no classList support	2 months ago
DataTransfer.js	Consistent license headers	4 months ago
Document.js	styling issues and support the filter as a function. IE requires it.	28 days ago
Element.js	don't fail if no classList support	2 months ago
FormData.js	Consistent license headers	4 months ago
HTMLAudioElement.js	Consistent license headers	4 months ago
HTMLCanvasElement.js	Consistent license headers	4 months ago
HTMLCollection.js	Consistent license headers	4 months ago
HTMLContentElement.js	Consistent license headers	4 months ago
HTMLElement.js	Consistent license headers	4 months ago
HTMLFormElement.js	Consistent license headers	4 months ago
HTMLImageElement.js	Consistent license headers	4 months ago
HTMLMediaElement.js	Consistent license headers	4 months ago
HTMLOptionElement.js	Consistent license headers	4 months ago
HTMLSelectElement.js	Consistent license headers	4 months ago
HTMLShadowElement.js	Consistent license headers	4 months ago
HTMLTableElement.js	Consistent license headers	4 months ago
HTMLTableRowElement.js	Consistent license headers	4 months ago
HTMLTableSectionElement.js	Consistent license headers	4 months ago
HTMLTemplateElement.js	Consistent license headers	4 months ago
HTMLUnknownElement.js	Consistent license headers	4 months ago
Node.js	adding gaps between isEqualNode and the function	28 days ago

<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.

```

551 var selectorRe = /([^{]*)({[\s\S]*?})/gim,
552 cssCommentRe = /\/*[^\s]*\*+([/*][^\s]*\*+)*\//gim,
553 // TODO(sorvell): remove either content or comment
554 cssCommentNextSelectorRe = /\/*\s*@polyfill ([^\s]*\*+([/*][^\s]*\*+)*\//)([^{]*){/gim,
555 cssContentNextSelectorRe = /polyfill-next-selector[^\s]*content\:[\s]*?["](.*)"["];\s*{([^{]*)}/gim,
556 // TODO(sorvell): remove either content or comment
557 cssCommentRuleRe = /\/*\s*@polyfill-rule([^\s]*\*+([/*][^\s]*\*+)*\//gim,
558 cssContentRuleRe = /(polyfill-rule)[^\s]*(content\:[\s]*["](.*)"["]);\s*[^\s]*}/gim,
559 // TODO(sorvell): remove either content or comment
560 cssCommentUnscopedRuleRe = /\/*\s*@polyfill-unscoped-rule([^\s]*\*+([/*][^\s]*\*+)*\//gim,
561 cssContentUnscopedRuleRe = /(polyfill-unscoped-rule)[^\s]*(content\:[\s]*["](.*)"["]);\s*[^\s]*}/gim,
562 cssPseudoRe = /::(x-[\s{,}*)/gim,
563 cssPartRe = /::part\(((^)*\)/gim,
564 // note: :host pre-processed to -shadowcsshost.
565 polyfillHost = '-shadowcsshost',
566 // note: :host-context pre-processed to -shadowcsshostcontext.
567 polyfillHostContext = '-shadowcsscontext',
568 parenSuffix = ')(?:\((( ' +
569     '(?:\(((^)([^\s]*)[^\s])?)+ ' +
570     ')\))?)?([^\s]*)';
571 var cssColonHostRe = new RegExp('(' + polyfillHost + parenSuffix, 'gim'),
572 cssColonHostContextRe = new RegExp('(' + polyfillHostContext + parenSuffix, 'gim'),
573 selectorReSuffix = '(>[\s~+\[\.,{:][\s\S]*)?$',
574 colonHostRe = /\:host/gim,
575 colonHostContextRe = /\:host-context/gim,
576 /* host name without combinator */
577 polyfillHostNoCombinator = polyfillHost + '-no-combinator',
578 polyfillHostRe = new RegExp(polyfillHost, 'gim'),
579 polyfillHostContextRe = new RegExp(polyfillHostContext, 'gim'),
580 shadowDOMSelectorsRe = [
581     /\^\/g,
582     /\^/g,
583     /\shadow\/g,
584     /\shadow-deep\/g,
585     /::shadow/g,
586     /\deep\/g,
587     /::content/g
588 ];

```

<https://github.com/webcomponents/webcomponentsjs/blob/9e5c7b527de95178a149db6f41b94bb4bfcd79db/src/ShadowCSS/ShadowCSS.js>

15 The intention here is to support only the styling features which can be  
16 relatively simply implemented. The goal is to allow users to avoid the  
17 most obvious pitfalls and do so without compromising performance significantly.  
18 For ShadowDOM styling that's not covered here, a set of best practices  
19 can be provided that should allow users to accomplish more complex styling.  
20



# 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.

# 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?

```
<script src="//cdnjs.com/2.7.0/moment.js"></script>
```

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

```
<script src="../../momentjs/moment.js"></script>
```

- 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.”*

# The catch

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





**Nicholas C. Zakas**

@slicknet



Following

Full video game engine with 3D rendering and real-world physics in a browser? Yes. Ability to style `<select>` dropdowns in a browser? No.



RETWEETS

**795**

FAVORITES

**343**



1:22 PM - 2 Jul 2014

# 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.

← ↻ 795 ★ 343 ⋮



**Oleg Aleynik** @oaleynik · Jul 2

.@slicknet @Web\_Components to the rescue! [youtube.com/watch?v=liK-7Q...](https://youtube.com/watch?v=liK-7Q...) (:

← ↻ 1 ★ 1 ⋮



**Jakob**  
@jakobo



+ Follow

.@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

← ↻ ★ ⋮

2:50 PM - 2 Jul 2014

# Good news: You can write custom elements that extend native form elements

```
1 <input is="red-input" />
```

HTML

1

```
1 var proto = Object.create( HTMLInputElement.prototype );
2 proto.createdCallback = function() {
3   this.style.backgroundColor = "red";
4   this.style.color = "white";
5   this.value = "Hello world";
6 }
7 document.registerElement( "red-input", {
8   prototype: proto,
9   extends: "input"
10 });
```

Hello world

**Bad news:** Custom elements don't give you any special control over the UI of native form elements.

mm / dd / yyyy



🔍 📱 Elements Network Sources Timeline Profiles »

▼ `<input type="date">`

▼ `#shadow-root (user-agent)`

▼ `<div pseudo="-webkit-datetime-edit" id="datetime-edit" datetimeformat="M/d/yy">`

▼ `<div pseudo="-webkit-datetime-edit-fields-wrapper">`

```
  <span role="spinbutton" aria-valuetext="blank" aria-valuemin="1" aria-valuemax="12" aria-help="Month" pseudo="-webkit-datetime-
```

html body input **#shadow-root**

- Chrome 31's shadow DOM implementation lets you replace the shadow root of `<input>` elements, which lets you alter the default UI.

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

HTML

```
1 input {  
2   height: 2em;  
3   width: 15em;  
4 }
```

```
1 var dateRoot = document.querySelector( "input" )  
2   .createShadowRoot();  
3  
4 $( "input" ).datepicker({  
5   dateFormat: "yy-mm-dd",  
6   onSelect: function( dateText ) {  
7     dateRoot.innerHTML = dateText;  
8   }  
9 });  
10
```

JavaScript

April 2015

Su	Mo	Tu	We	Th	Fr	Sa
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

- **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)<sup>1</sup>.

1: <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 issues<sup>1</sup>
- Provides more value to frameworks than individual apps
- My recommendation: **Hold off**

1: <https://github.com/w3c/webcomponents/wiki/Shadow-DOM:-Contentious-Bits> & <https://lists.w3.org/Archives/Public/public-webapps/2015AprJun/0052.html>

# HTML Imports

- Chrome has the only stable implementation
  - Firefox is holding off<sup>1</sup>
- No good solution to manage third-party dependencies
- My recommendation: **Hold off**

1: <https://hacks.mozilla.org/2014/12/mozilla-and-web-components/>

# <template>

- 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/>
  - Evergreen browsers, IE9+, Android 4+
- <https://github.com/WebReflection/document-register-element>
  - Evergreen browsers, **IE8+**, iOS 5+, Android 2.2+
  - **2.5K!**



# Custom elements API improvements

```
<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 ...

[← Telerik Kendo UI Feedback](#)

97

votes

Vote

## 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](#)

65

votes

Vote

## 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...](#)

I suggest you ...

← [Telerik Kendo UI Feedback](#)

**3**

votes

Vote

## 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**

- <https://github.com/telerik/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.

 Tweet 34


 +1 658

[Getting Started](#)[Dropdown List](#)[Combobox](#)[Number Picker](#)[Multiselect](#)[SelectList](#)[Calendar](#)

# Getting Started

---

current version 2.4.0

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
  - <http://developer.telerik.com/featured/web-components-ready-production/>
- 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>