CROSS-FRAMEWOR COMPONENTS

ArcGIS Portal App

REUSABLE UI COMPONEN

Currently we use Dojo's Dijit library to create re
components in both the JS API and Por

WHY DIJITS?

- Cross-browser
- Accessible
- Localizable
- Themeable

PROBLEMS WITH DIJITS

- Lots of complexity
- No future in Dojo 2.0
- Needs wrapping to use with other to
- Hard to style
- Browsers have standardized

How can we build modular components compati
wide range of build tools, module systems a
frameworks being used at Esri?

- Angular 1.0 - Developers Site, Open Data Adr
- Angular 2.0 - Insights
- Backbone - Open Data
- Ember - Open Data, Operations Dashboard, W
- Dojo - ArcGIS Online/Portal App
- jQuery - My Stories

WEB COMPONENTS

New web standard for building reusable UI co

- Custom Elements
- Shadow DOM
- Templates
- HTMl Imports

CUSTOM ELEMENTS

Create custom HTML tags like `<item-rati`
`<share-button>`. These custom elements b
like `<a>`, `<button>` or `<form>`.

SHADOW DOM

Create fragments of DOM to separate compone
and scripts from the rest of the page. This make
isolate components.

TEMPLATES

Create reusable templates that can be use in co

HTML IMPORTS

Combine HTML, CSS, and JavaScript into a sing
can be imported into existing HTML doc

THE STATE OF WEB COMPON

Feature	Chrome	Firefox	Safar
Custom Elements	✓	✓ (Flag)	?
Shadow DOM	✓	✓ (Flag)	?
Templates	✓	✓	✓
HTML Imports	✓	X	?

Are We Componentized Yet?

POLYFILLING WEB COMPON

Use existing APIs to add support for future stand
for all browsers back to IE 9.

PRACTICAL WEB COMPONE

- Pollyfilled Shadow DOM cannot encapsula
- Templates are only useful with HTML Imp
- Firefox won't support HTML Imports

CUSTOM ELEMENTS

Even without Shadow DOM, Templates, and HTM
Custom Elements are still amazingly useful. Co
simple component to rate an item in Por

A SIMPLE ITEM RATING COMP

```
<item-rating
    itemid="30e5fe3149c34df1ba922e6f5bbf808f"
    numratings="6"
    rating="4.25"
></item-rating>
```

```
var itemRating = new ItemRating({
    rating: 4.25,
    numratings: 6,
    itemid: '30e5fe3149c34df1ba922e6f5bbf808f'
});

document.body.appendChild(itemRating);
```

APPS WITH COMPONENT

```
<h1>My Item</h1>

<arcgis-item-rating
    itemid="30e5fe3149c34df1ba922e6f5bbf808f"
    numratings="6"
    rating="4.25"></arcgis-item-rating>

<a onclick="document.getElementById('#share-modal').open(

<calcite-modal id="#share-modal">
    <h2>Share</h2>
    <arcgis-share-button
        itemid="30e5fe3149c34df1ba922e6f5bbf808f"></arcgi
<calcite-modal>
```

INSIDE A CUSTOM ELEME

```
class ItemRating extends HTMLElement {
  createdCallback () {
    // called when the element is first created
  }

  attachedCallback () {
        // called whenever an element is added to the DOM
  }

  detachedCallback () {
    // called when the element is removed from the DOM
  }

  attributeChangedCallback (attribute, oldValue, newValue
        // called whenever an attribute changes on an ele
    }
```

Full Source

FRAMEWORK COMPATIBIL

- Declarative API
- Programatic API
- Backbone
- Angular 1.0
- Angular 2.0
- Ember 2.0
- Aurelia

JavaScript libraries like Dojo and jQuery woul
programatic or declarative APIs.

FUTURE PROOFING

- Move to an app framework in the future
- All app frameworks need to work with DOM
- Other teams can use components from port

CHALLENGES

- Building and Compiling
- Localization
- Accessibility
- Style Collisions

BUILDING AND COMPILIN

Require adding a compiler to transform the ES 2
to current ES 5. We could use either Babel or Typ
this and run it before the main Dojo build

LOCALIZATION - SHORT TE

Use the existing localization tools from D

```
import i18n from 'dojo/i18n!arcgisonline/nls';

class ItemRating extends HTMLElement {
    // ... use ItemRating.i18n
}

const ItemRating.i18n = i18n;
```

LOCALIZATION - LONG TE

Bundle localizations for each component sepa
other teams do not have to rely on Dojo to wo
discussed in more detail with the team

ACCESSIBILITY

Since web components are simply DOM elemen
use the standard ARIA best practices to ensure
accessible.

STYLE COLLISIONS

We may run into situations where styles from
bleed into each other.

1. Assume Calcite Web will be used
2. Keep component structure simple
3. Namespace all CSS selectors

IMPLEMENTATION

How do components interact with the Port

1. Requests item details from API
2. Builds components
3. Listens for events like `rateitem`
4. App makes API calls
5. Updates `<item-rating>` attribut

IMPLEMENTATION

As much as possible components should not kr
the JS API or the REST APIs.

PROPOSED PLAN

Start implementing new item page design using

- Add compilers to build tools
- Break down items page into compone
- Begin building components
- Start to wire components to the API

LONG TERM BENEFITS

- Easy to use an app framework in the future if n
- Components can be shared with applications o
  Portal App
- App becomes highly modular and easy to reas