ArcGIS Experience Builder Development Considerations

David Howes, Ph.D.
David Howes, LLC

Willamette Valley GIS Users Group
February 21st, 2024
David Howes, Ph.D.

• Education
  • B.Sc. (Hons) in Geography - University of Salford, England
  • M.Sc. in Geographic Information Systems - University of Edinburgh, Scotland
  • Ph.D. in Geomorphology - State University of New York at Buffalo, New York

• > 30 years in GIS

• Specialty: GIS tools, processes, and supporting infrastructure

• Established
  • David Howes, LLC in 2012
  • GISPД.com in 2014
Session Purpose

• Review ArcGIS Experience Builder customization options

• Provide helpful information for developers and non-developers

• Encourage move from ArcGIS Web AppBuilder to ArcGIS Experience Builder
Contents

• Experience Builder overview
• Development options
• Widget development
• Traditional development
• Moving from Web AppBuilder
• Discussion
• Closing considerations
Acknowledgements

• Joe Rhodes - CIVICLENS, Missoula, Montana
  http://civiclens.com

• Eric McAvoy - Polk County, Oregon
  https://www.co.polk.or.us/ms/gis/geographic-information-systems-gis

• David Jacobus - City of Albany, Oregon
  https://albanyoregon.gov/gis
Experience Builder Overview
ArcGIS Experience Builder

"a highly configurable solution for building compelling web apps without writing code"

Why Experience Builder?

- ArcGIS Maps SDK for JavaScript 4.x
- Mobile friendly
- Modern capabilities, e.g., Survey123 widget
- Multimedia technology

Web AppBuilder Retirement

ArcGIS Web AppBuilder Roadmap for Retirement

Announcements
February 27, 2023

ArcGIS Web App Builder is retiring, and we want to share the retirement timelines and details with you. While your ArcGIS Web AppBuilder apps that you have built will continue to work, Esri’s recommended path for taking advantage of new capabilities in ArcGIS as well as for staying abreast of the changes in browser technology is to migrate applications to ArcGIS Experience Builder. Here are the timelines and details:

Developer Edition

The developer edition will retire in July 2024, coinciding with the retirement of ArcGIS API for JavaScript version 3.x.


GISPD.com
GIS Professional Development
Development Options
Development Options

• Configuration
  • Change settings and assemble parts to create an application and control its look and capabilities
  • E.g.,
    • Change the theme
    • Add new functionality via existing widgets

• Customization
  • Develop and add widgets
  • Alter application by adding, removing, or modifying code
# Development Platform

<table>
<thead>
<tr>
<th></th>
<th>Use Custom Widgets</th>
<th>Customization &amp; Widget Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArcGIS Online</td>
<td>Yes via marketplace?</td>
<td>No</td>
</tr>
<tr>
<td>ArcGIS Enterprise</td>
<td>Yes (10.8.1+)</td>
<td>No</td>
</tr>
<tr>
<td>Developer Edition</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Experience Builder Developer Edition

• Standalone version of Experience Builder allowing customization

• Download from Developer Edition Landing Page
  https://developers.arcgis.com/experience-builder

• Typically one or two months behind Online version in the Esri development release cycle
Hosting

• Download app from Developer Edition

• Host in any web server
Widget Development
Map Widget, Settings
Select Features Widget, Settings
Select Features Widget
Select Features
Selection Result
Anatomy of a Web Page

• Page is made of objects.

• The objects are arranged in a hierarchy

• That hierarchy is referred to as the Document Object Model (DOM)
  
HTML

- HTML (HyperText Markup Language) is the language used to define the meaning and structure of web pages

- Within a page, many of the objects are HTML elements

- E.g., table cell element

  <td>Cell A</td>
Table Element Hierarchy

The `td` element resides within a table element hierarchy

```html
<table>
  <tr>
    <td>Cell A</td>
    <td>Cell B</td>
  </tr>
</table>
```
Core Technology

• HTML - to provide structure

• Cascading Style Sheets (CSS) - to control the look of the page

• JavaScript - to provide interactivity
```html
<div class="normal-text-div">
    <table cellpadding="0" cellspacing="0">
        <tbody>
            <tr>
                <td>
                    <div id="selectButton" data-dojo-attach-point="btnSelectFeatures"
                        data-dojo-attach-event="onclick:_onBtnSelectFeaturesClicked" class="jimu-btn">
                        ${nls.buttonTextSelectFeatures}
                    </div>
                </td>
                <td>
                    <div id="clearButton" data-dojo-attach-point="btnClearSelection"
                        data-dojo-attach-event="onclick:_onBtnClearSelectionClicked" class="jimu-btn">
                        ${nls.buttonTextClearSelection}
                    </div>
                </td>
            </tr>
        </tbody>
    </table>
</div>
```
Web AppBuilder Widget: JavaScript, Modules

```javascript
define(
    // Specify required modules (dependencies)
    [
        "dojo/_base/declare",
        "jimu/BaseWidget",
        "dijit/_TemplatedMixin",
        "dijit/_WidgetsInTemplateMixin",
        "esri/symbols/SimpleFillSymbol",
        "esri/symbols/SimpleLineSymbol",
        "esri/tasks/QueryTask",
        "esri/tasks/query",
        "esri/toolbars/draw",
        "esri/Color",
        "esri/graphic",
        "dojo/_base/lang",
        "dojo/_base/array",
        "dojo/dom",
        "dojo/dom-style"
    ],

    // Provide names (aliases) for modules
    function () {
        declare,
        BaseWidget,
        _TemplatedMixin,
        _WidgetsInTemplateMixin,
        SimpleFillSymbol,
    }
```
Component-Based Development

• Create element-like objects referred to as components

• Web application is a hierarchy of components

• Widgets are components that sit within the hierarchy and may be composed of other components
Experience Builder Component - Basic Widget

• Created using React
  JavaScript library for building user interfaces
  https://reactjs.org

• Basic widget from Esri:

```javascript
import { React, AllWidgetProps } from "jimu-core";

function Widget(props: AllWidgetProps<any>) {
  return <div className="widget-starter jimu-widget">Basic widget...</div>;
}

export default Widget;
```

• The component is defined by a JavaScript function
• The function returns a single HTML-like element
• The mix of JavaScript and HTML is called JavaScript XML (JSX)
Experience Builder Technology (1)

- React
  JavaScript library for building user interfaces
  https://reactjs.org

- TypeScript
  Language that builds on JavaScript by adding static type definitions
  https://www.typescriptlang.org

- ArcGIS Map SDK for JavaScript 4.x
  Esri's primary JavaScript library
  https://developers.arcgis.com/javascript
Experience Builder Technology (2)

- npm
  A JavaScript package manager (Node Package Manager)
  https://www.npmjs.com

- Redux
  JavaScript library for managing application state (i.e., data)
  https://redux.js.org

- WebPack
  Module bundler
  https://webpack.js.org
Experience Builder Technology (3)

- Emotion
  Library for writing CSS styles with JavaScript
  [https://emotion.sh/docs/introduction](https://emotion.sh/docs/introduction)

- Styled Components (CSS)
  CSS-in-JavaScript library that bridges the gap between components and styling
  [https://styled-components.com](https://styled-components.com)
TypeScript: Static Typing

- JavaScript
  
  Any valid object can be assigned to a variable at any time

  ```javascript
  var x = 3;
  x = "something";
  x = true;
  ```

- TypeScript

  ```javascript
  let x: number = 3;

  let x: number
  Type 'string' is not assignable to type 'number'. ts(2322)

  View Problem (Alt+F8)  No quick fixes available
  x = "something";
  ```
Select Features Widget: Imports

```typescript
/** @jsx jsx */

// Import statements.
import { React, jsx, css, styled, type AllWidgetProps } from 'jimu-core';
import { type JimuMapView, JimuMapViewComponent } from 'jimu-arcgis';
import FeatureLayer from 'esri/layers/FeatureLayer';
import SimpleLineSymbol from 'esri/symbols/SimpleLineSymbol';
import SimpleFillSymbol from 'esri/symbols/SimpleFillSymbol';
import GraphicsLayer from 'esri/layers/GraphicsLayer';
import FeatureSet from 'esri/rest/support/FeatureSet';
import Graphic from 'esri/Graphic';
import SketchViewModel from 'esri/widgets/Sketch/SketchViewModel';

// Widget config object.
import { IMConfig } from '../*/config';

// Widget messages (strings) object.
import defaultMessages from './translations/default';

// React hooks.
const { useState, useEffect } = React;

function Widget(props: AllWidgetProps<IMConfig>) {
  // Widget component.
  const useMapWidgetId: string = props.useMapWidgetIds?.[0];
```

React

ArcGIS Maps SDK for JavaScript 4.x

TypeScript
function selectFeatures() {
    // Disable layer popups.
    featureLayer.popupEnabled = false;

    // Create a new SketchViewModel.
    const graphicsLayer: GraphicsLayer = new GraphicsLayer();
    const sketchViewModel: SketchViewModel =
       new SketchViewModel(
            layer: graphicsLayer,
            view: jimuMapView.view,
            polygonSymbol: selectionSimpleFillSymbol
       );

    // Listen to the SketchViewModel's create event.
    sketchViewModel.on('create', (function (event: any) {
        // Check the event state.
        if (event.state === 'complete') {
            // Create a query options object using the user-defined geometry.
            const queryOptions: IQueryOptions = {
                where: '',
                geometry: event.graphic.geometry,
                returnGeometry: true
            };

            // Query the features.
        }
    });
}
Select Features Widget: Return

return {
  // Render the widget UI.
  <div className='widget-select-features jimu-widget'>
    <div css={headerStyleLiteral}>{defaultMessages.header}</div>
    
    <StyledButton 
      onClick={handleSelectFeaturesButtonClick} 
      disabled={!configured} 
      >{defaultMessages.btnSelectFeaturesCaption} 
    </StyledButton>
    &nbsp;
    <StyledButton 
      onClick={handleClearSelectionButtonClick} 
      disabled={!configured} 
      >{defaultMessages.btnClearSelectionCaption} 
    </StyledButton>

  </div>

  {useMapWidgetId && 
    <JimuMapViewComponent 
      useMapWidgetId={useMapWidgetId} 
      activeViewChange={setJimuMapView} />
  }

}
styled-components

Select Features Widget: Styles

```javascript
// Create a style literal for the header.

const headerStyleLiteral: string = css`
  color: ${props.theme.colors.black};
  font-size: 1rem;
  font-weight: bold;
```

// Create a styled button.

const StyledButton: any = styled.button`

${configured && css`
  color: white;
  background-color: ${props.theme.colors.primary};
```

${!configured && css`
  color: white;
  background-color: ${props.theme.colors.secondary};
```

return (  
  // Render the widget UI.

  <div className='widget-select-features jimu-widget'>
    <div css={headerStyleLiteral}>{defaultMessages.header}</div>
  
```
Experience Builder Resources

• ArcGIS Experience Builder Developer Edition landing page
  https://developers.arcgis.com/experience-builder

• GitHub samples repository
  https://github.com/Esri/arcgis-experience-builder-sdk-resources

• Esri community
  https://community.esri.com/community/arcgis-experience-builder
Esri TypeScript Resources

• API Guide
  * TypeScript - Setting up your development environment*
  https://developers.arcgis.com/javascript/latest/guide/typescript-setup

• GitHub repository
  https://github.com/Esri/jsapi-resources/tree/main/typescript

• Developer Summit videos
  * ArcGIS API for JavaScript: Using TypeScript*
  https://www.youtube.com/watch?v=m385zKppkUs
Traditional Development
# Table Widget

![Table Widget Image]

The image shows a table widget with the following columns:

- **OBJECTID**
- **STATEFP**
- **COUNTYFP**
- **COUNTRYS**
- **GEOID**

The table contains data for 20 rows, each representing a different object with corresponding values in the mentioned columns.

---

**Source:** GISPD.com - GIS Professional Development
Table Widget Style Configuration

- Set column colors according to configuration values

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Column Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATEFP</td>
<td>orange</td>
</tr>
<tr>
<td>COUNTYFP</td>
<td>aqua</td>
</tr>
<tr>
<td>COUNTYNS</td>
<td>gray</td>
</tr>
</tbody>
</table>

- Allow for row selection
### Table Widget Styling

<table>
<thead>
<tr>
<th>OBJECTID</th>
<th>STATEFP</th>
<th>COUNTYFP</th>
<th>COUNTYNS</th>
<th>GEOID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>31</td>
<td>039</td>
<td>00835841</td>
<td>31039</td>
</tr>
<tr>
<td>2</td>
<td>53</td>
<td>069</td>
<td>01513275</td>
<td>53069</td>
</tr>
<tr>
<td>3</td>
<td>35</td>
<td>011</td>
<td>00933054</td>
<td>35011</td>
</tr>
<tr>
<td>4</td>
<td>31</td>
<td>109</td>
<td>00835876</td>
<td>31109</td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>129</td>
<td>00835886</td>
<td>31129</td>
</tr>
<tr>
<td>6</td>
<td>72</td>
<td>085</td>
<td>01804523</td>
<td>72085</td>
</tr>
<tr>
<td>7</td>
<td>46</td>
<td>099</td>
<td>01265772</td>
<td>46099</td>
</tr>
<tr>
<td>8</td>
<td>48</td>
<td>327</td>
<td>01383949</td>
<td>48327</td>
</tr>
<tr>
<td>9</td>
<td>06</td>
<td>091</td>
<td>00277310</td>
<td>06091</td>
</tr>
<tr>
<td>10</td>
<td>21</td>
<td>053</td>
<td>00516873</td>
<td>21053</td>
</tr>
<tr>
<td>11</td>
<td>39</td>
<td>063</td>
<td>01074044</td>
<td>39063</td>
</tr>
<tr>
<td>12</td>
<td>48</td>
<td>189</td>
<td>01383880</td>
<td>48189</td>
</tr>
<tr>
<td>13</td>
<td>01</td>
<td>027</td>
<td>00181539</td>
<td>01027</td>
</tr>
<tr>
<td>14</td>
<td>48</td>
<td>011</td>
<td>01383791</td>
<td>48011</td>
</tr>
<tr>
<td>15</td>
<td>39</td>
<td>003</td>
<td>01074015</td>
<td>39003</td>
</tr>
<tr>
<td>16</td>
<td>13</td>
<td>189</td>
<td>00348794</td>
<td>13189</td>
</tr>
<tr>
<td>17</td>
<td>55</td>
<td>111</td>
<td>01581115</td>
<td>55111</td>
</tr>
<tr>
<td>18</td>
<td>05</td>
<td>137</td>
<td>00699902</td>
<td>05137</td>
</tr>
<tr>
<td>19</td>
<td>41</td>
<td>063</td>
<td>01155135</td>
<td>41063</td>
</tr>
</tbody>
</table>
Custom Table Widget?

- Table widget relies on an Esri FeatureTable object in the ArcGIS Maps SDK for JavaScript 4.x

- Ideally
  - Install the SDK locally
  - Create a custom copy of the FeatureTable code with adjusted cell styling
  - Use the updated code in a copy of the Table widget

- In reality
  FeatureTable code is minified, so it's not customizable
define("require ../_rollupPluginBabelHelpers ../_chunks/tslib.es6
../_int1 ../_core/deprecate ../_core/HandleOwner ../_core/Logger
../_core/reactiveUtils ../_core/accessorSupport/decorators/property
../_core/accessorSupport/decorators/cast ../_core/arrayUtils ../_core/has
../_core/accessorSupport/decorators/subclass ../_Widget
../FeatureTable/FeatureTableViewModel ../_FeatureTable/Grid/support/ButtonMenu
../FeatureTable/Grid/support/ButtonMenuItem ../_support/componentsUtils
../support/Heading ../_support/widgetUtils ../_support/decorators/messageBundle
../support/jsxFactory ../_int1/substitute".split(" "),
function(w,u,e,d,y,D,E,r,f,F,M,N,G,H,I,J,n,K,L,O,x,h,z){const A={header:!0,
menu:!0,menuItemItems:{clearSelection:!0,refreshData:!0,toggleColumns:!0,
selectedRecordsShowAllToggle:!0,selectedRecordsShowSelectedToggle:!0,
zoomToSelection:!0,deleteSelection:!0},selectionColumn:!0,columnMenus:!0},B=
E.getLogger("esri.widgets.FeatureTable");d=function(C){function v(a,b){a=C.
call(this,a,b)||this; a._prompt=null; a.menu=null; a.menuConfig=null; a.
viewModel=new I; a.visibleElements={...A}; a._showDeletePrompt=a.
_showDeletePrompt.bind(u._assertThisInitialized(a));
a._onDeleteSelectionClick=a._onDeleteSelectionClick.bind(u.
_assertThisInitialized(a));return a}u._inherits(v,C);var c=v.prototype;c.
initialize=function(){this.handles.add([r.on(()=>this.viewModel.columns,
"change",())=>this._syncMenuConfig(),r.on(()=>this.viewModel.activeFilters,
"change",())=>this._syncMenuConfig()],g
Element Attributes

• Elements have data attached to them for various purposes

  <td width="300" height="20">Cell A</td>

• Can use browser developer tools to inspect elements
Goal

Use the attributes to find the Table widget td (i.e., cell) elements and apply styles to them
Table Web Component

• The Table widget uses a reusable web component

• The web component includes a template grid consisting of placeholders called "slots" that can be filled with markup

• The template is loaded with HTML cell elements and then refreshed as the user scrolls through the rows or filters the data
Template Slot Element

```html
<vaadin-grid-cell-content slot="vaadin-grid-cell-content-88" title="2">2</vaadin-grid-cell-content>
```

```javascript
const slot = $0
```
```html
<td id="vaadin-grid-cell-88" tabindex="0" role="gridcell" part="cell body-cell" first-column reorder-status="undefined" aria-selected="true" class="OBJECTID" style="width: 200px; flex-grow: 1; order: 20000000;" flex

</td>
```
Selected Cell - aria-selected

```html
<td id="vaadin-grid-cell-88" tabindex="0"
role="gridcell" part="cell body-cell"
first-column reorder-status="undefined"
aria-selected="true" class="OBJECTID"
style="width: 200px; flex-grow: 1; order: 20000000;">
  flex
</td>
```

ARIA (Accessible Rich Internet Applications)
"set of roles and attributes that define ways to make web content and web applications (especially those developed with JavaScript) more accessible to people with disabilities."

Column Cell Style Class

Create an HTML style class

• E.g.,

```html
.tableCol_1[aria-selected="false"] { background-color: orange }
```

• Add the class to the DOM via a style element
Style Class Application

• Loop over all `td` (i.e., cell) elements in the web component template

• Identify those in the appropriate column

• Add the class name to the list of classes for the column cells
Style Columns

<table>
<thead>
<tr>
<th>OBJECTID</th>
<th>STATEFP</th>
<th>COUNTYFP</th>
<th>COUNTRYNS</th>
<th>GEOID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>31</td>
<td>039</td>
<td>00835841</td>
<td>31039</td>
</tr>
<tr>
<td>2</td>
<td>53</td>
<td>069</td>
<td>01513275</td>
<td>53069</td>
</tr>
<tr>
<td>3</td>
<td>35</td>
<td>011</td>
<td>00933054</td>
<td>35011</td>
</tr>
<tr>
<td>4</td>
<td>31</td>
<td>109</td>
<td>00835876</td>
<td>31109</td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>129</td>
<td>00835886</td>
<td>31129</td>
</tr>
<tr>
<td>6</td>
<td>72</td>
<td>085</td>
<td>01804523</td>
<td>72085</td>
</tr>
<tr>
<td>7</td>
<td>46</td>
<td>099</td>
<td>01265772</td>
<td>46099</td>
</tr>
<tr>
<td>8</td>
<td>48</td>
<td>327</td>
<td>01383949</td>
<td>48327</td>
</tr>
<tr>
<td>9</td>
<td>06</td>
<td>091</td>
<td>00277310</td>
<td>06091</td>
</tr>
<tr>
<td>10</td>
<td>21</td>
<td>053</td>
<td>00516873</td>
<td>21053</td>
</tr>
<tr>
<td>11</td>
<td>39</td>
<td>063</td>
<td>01074044</td>
<td>39063</td>
</tr>
<tr>
<td>12</td>
<td>48</td>
<td>189</td>
<td>01383880</td>
<td>48189</td>
</tr>
<tr>
<td>13</td>
<td>01</td>
<td>027</td>
<td>00161539</td>
<td>01027</td>
</tr>
<tr>
<td>14</td>
<td>48</td>
<td>011</td>
<td>01383791</td>
<td>48011</td>
</tr>
<tr>
<td>15</td>
<td>39</td>
<td>003</td>
<td>01074015</td>
<td>39003</td>
</tr>
<tr>
<td>16</td>
<td>13</td>
<td>189</td>
<td>00348794</td>
<td>13189</td>
</tr>
<tr>
<td>17</td>
<td>55</td>
<td>111</td>
<td>01581115</td>
<td>55111</td>
</tr>
<tr>
<td>18</td>
<td>05</td>
<td>137</td>
<td>00069902</td>
<td>05137</td>
</tr>
<tr>
<td>19</td>
<td>41</td>
<td>063</td>
<td>01155135</td>
<td>41063</td>
</tr>
</tbody>
</table>
Required Code

• Two files
  • JavaScript processing code
  • JavaScript object containing config information

• Each file is referenced via a script element in the application index.html file
Moving from Web AppBuilder
Moving from Web AppBuilder (1)

- Web AppBuilder retirement - July 2024

- If you haven't already done so, start making your move to Experience Builder as soon as possible

- Experience Builder has reached parity with Web AppBuilder, but the functional correspondence is not necessarily one-to-one
  It may be tricky to find what you need
Moving from Web AppBuilder (2)

• Dojo may be seen as a security threat

• Google is deprecating functionality related to Dojo (the framework supporting widgets)

• If required, download a copy of the ArcGIS Maps SDK for JavaScript now
Starting from Scratch

- Web AppBuilder templates are now provided in Experience Builder
- You can't just import and translate a Web AppBuilder application or Web AppBuilder widgets into Experience Builder
- You have to build new applications and widgets from scratch
- Take the time to research requirements thoroughly
Discussion
Closing Considerations
Closing Considerations

• Customizing Experience Builder is more challenging than customizing Web AppBuilder

• If you're not already doing so, start making a move from Web AppBuilder to Experience Builder now

• Research options before embarking on customization
Thanks for Participating

For further information and assistance:

info@gispd.com

Welcome to GISPD.com
Our mission. We endeavor to support GIS professionals by providing informative resources on our website, organizing and participating in professional gatherings and conducting training events.

Oct 13, 2020
ArcGIS Experience Builder Widget Development: An Introduction for the GIS Layperson
Presentation to be given online as part of Northwest GIS 2020
David Howes, David Howes, LLC
Full details