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
  - GISPD.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

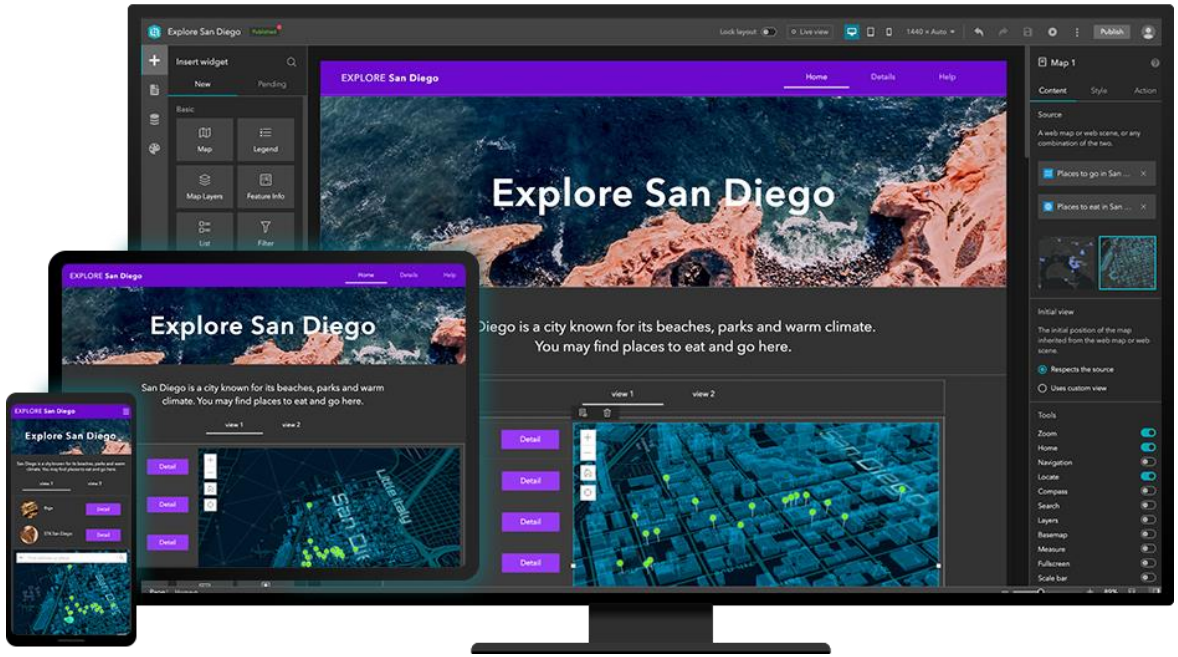
- Joe Rhodes - CIVICLENS, Missoula, Montana
  http://civiclens.com

- 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"

https://www.esri.com/en-us/arcgis/products/arcgis-experience-builder/overview

Why Experience Builder?

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

https://www.esri.com/en-us/arcgis/products/arcgis-experience-builder/overview

Web AppBuilder Retirement

ArcGIS Web AppBuilder Roadmap for Retirement

Announcements
February 27, 2023

Jianxia Song
Eva Moyer

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.

https://www.esri.com/arcgis-blog/products/web-appbuilder/announcements/arcgis-web-appbuilder-roadmap-for-retirement

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

	Use Custom Widgets	Customization & Widget Development
ArcGIS Online	Yes via marketplace?	No
ArcGIS Enterprise	Yes (10.8.1+)	No
Developer Edition	Yes	Yes

Experience Builder Developer Edition

- Standalone version of Experience Builder allowing customization

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

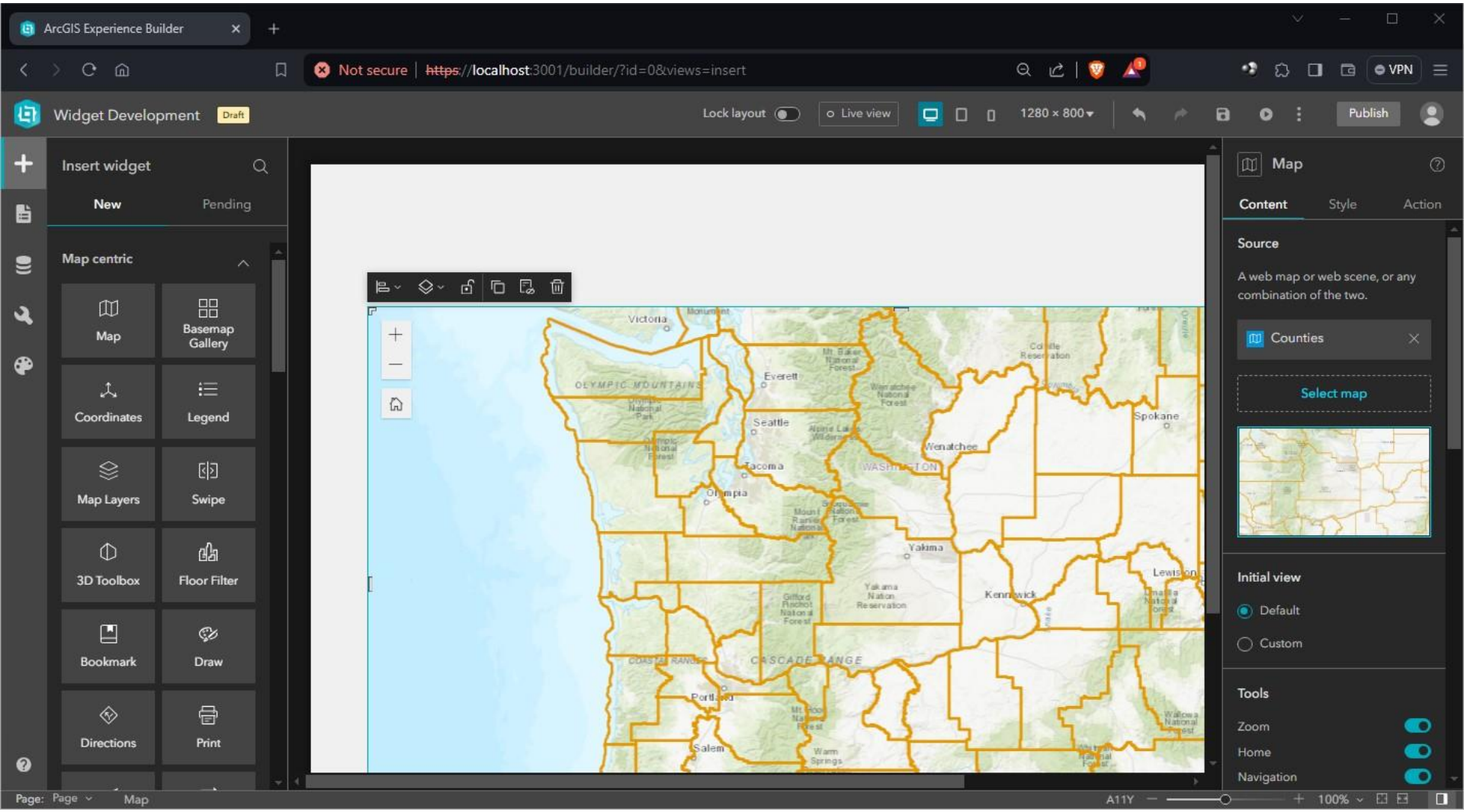
- Typically one or two months behind Online version in the Esri development release cycle

Hosting

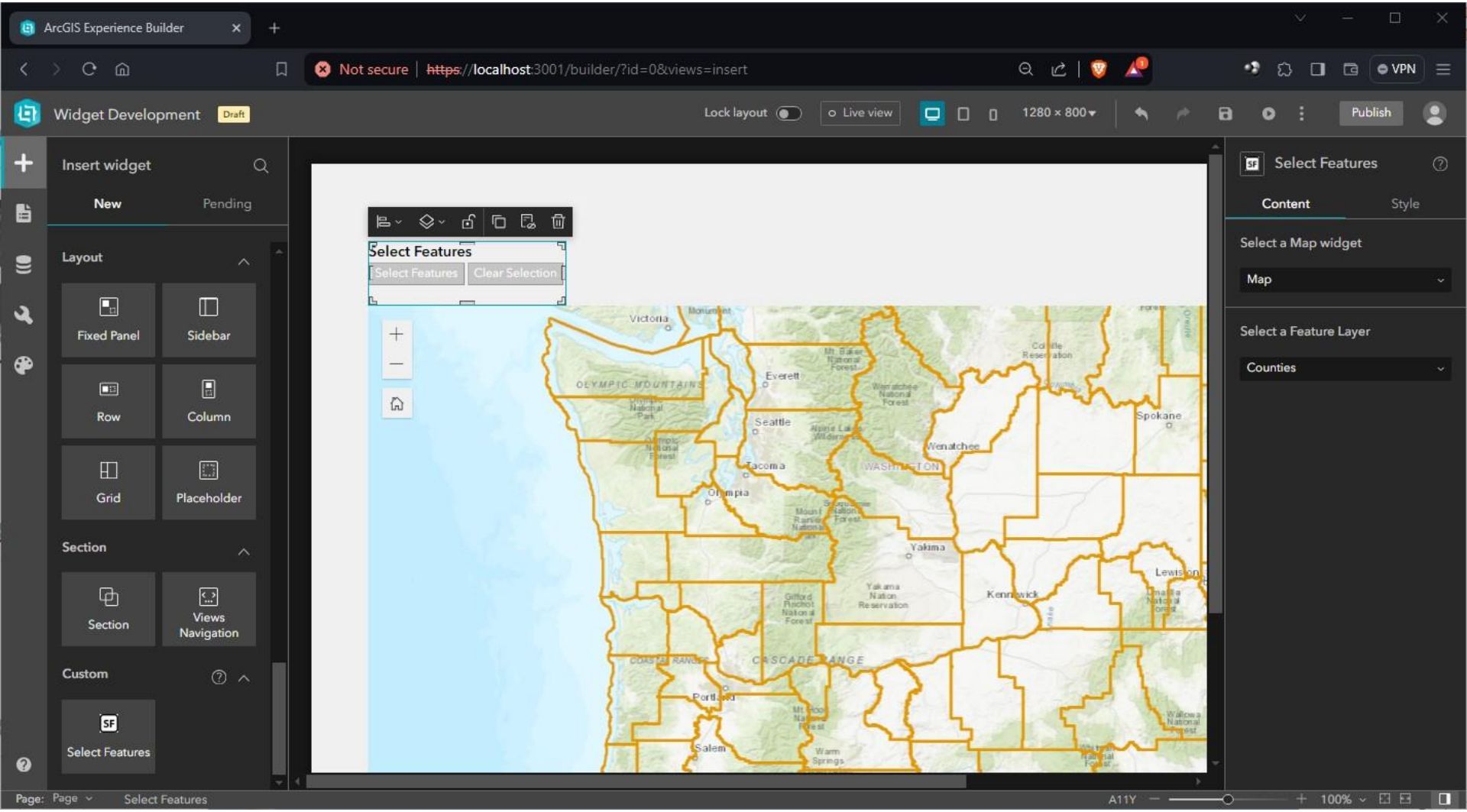
- Download app from Developer Edition
- Host in any web server

Widget Development

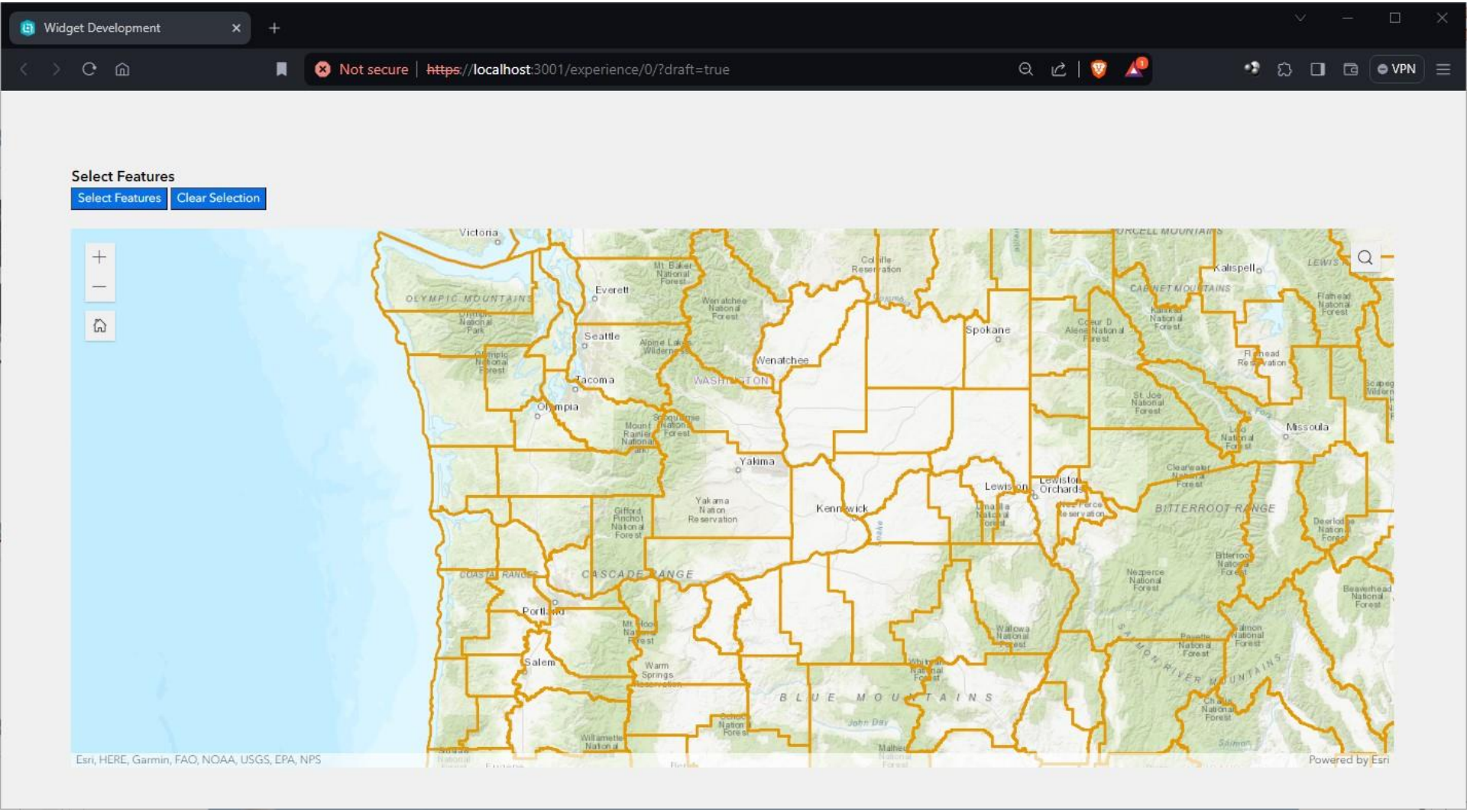
Map Widget, Settings



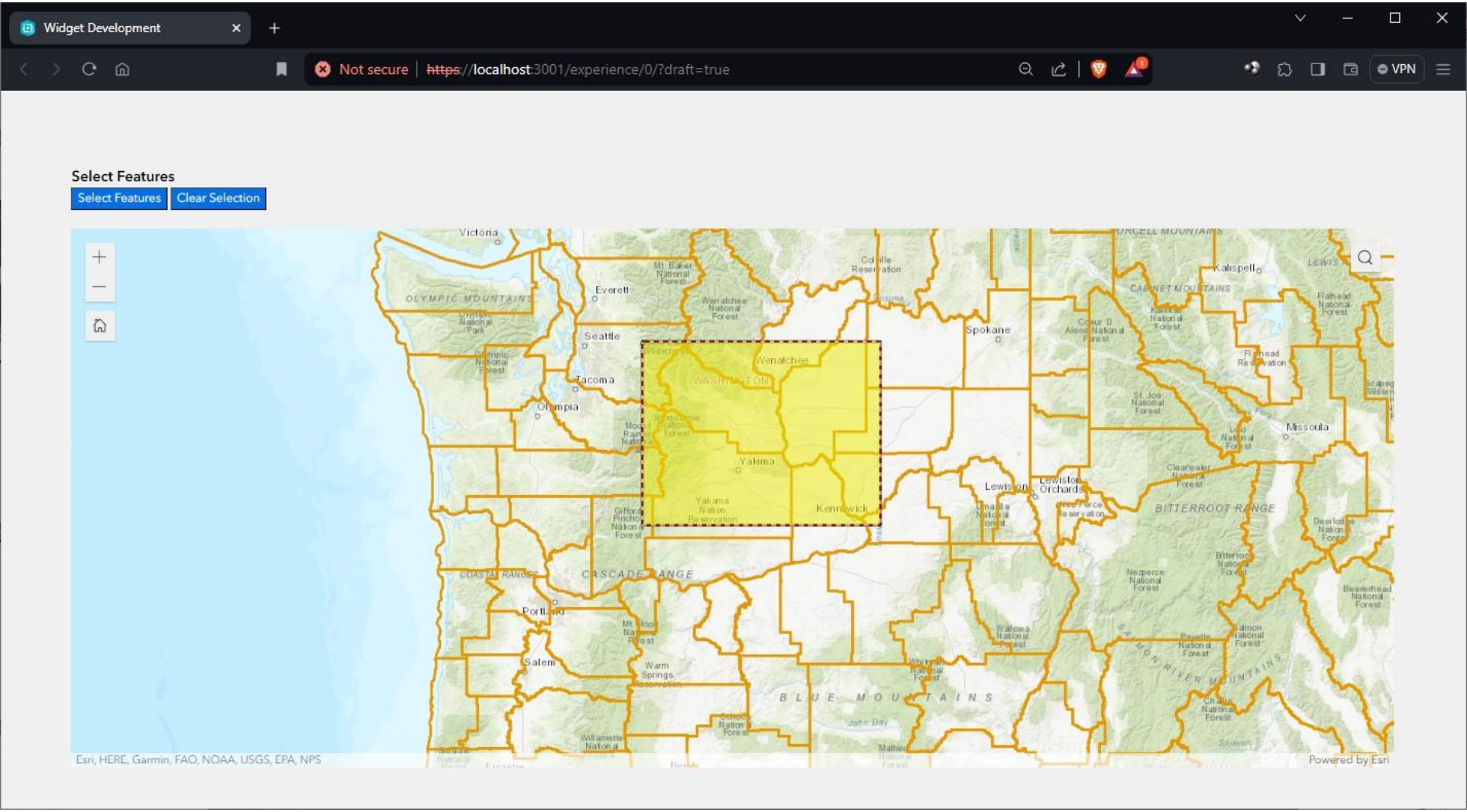
Select Features Widget, Settings



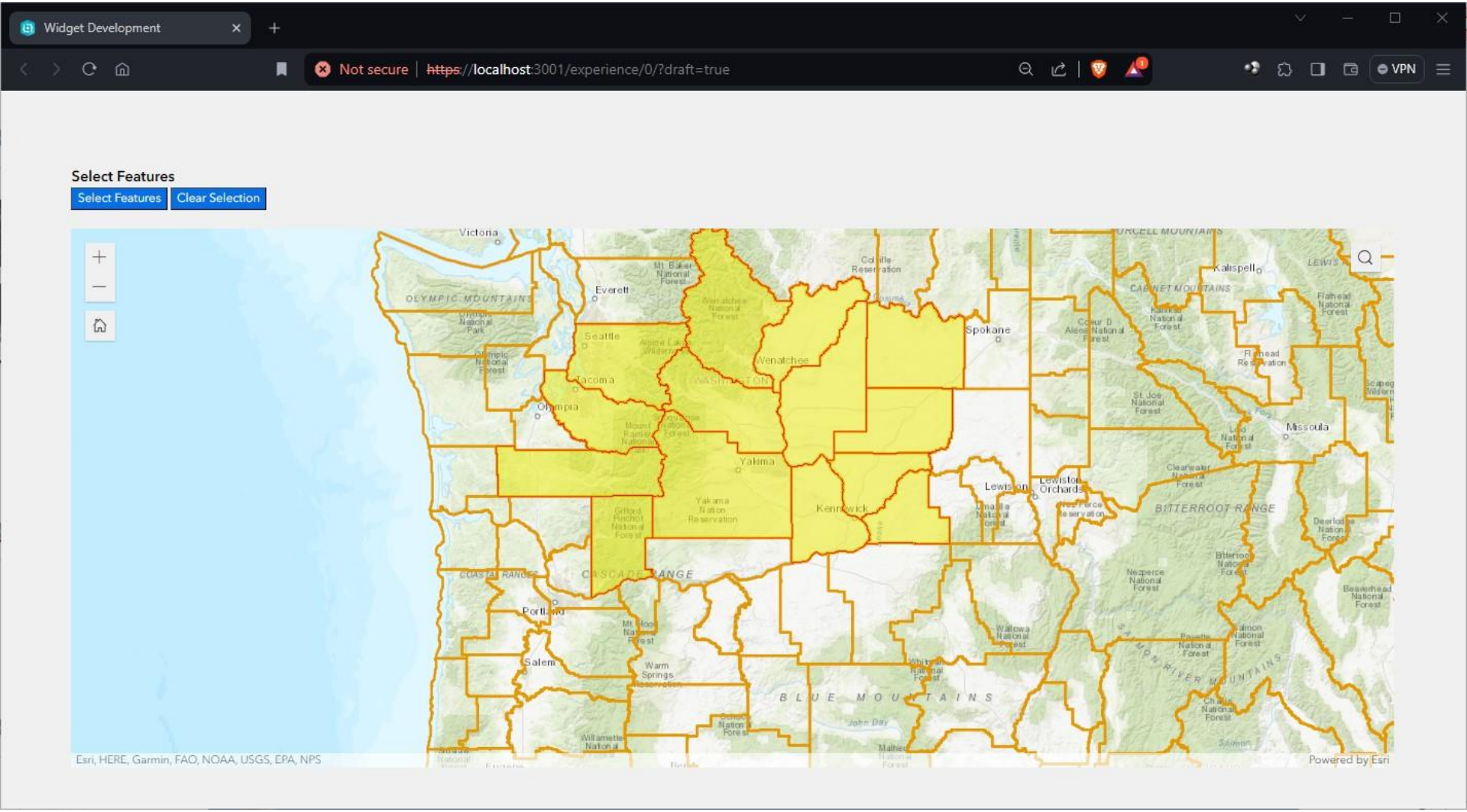
Select Features Widget



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)

  https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

HTML

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

  https://developer.mozilla.org/en-US/docs/Web/HTML

- 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

```
<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

Web AppBuilder Widget: HTML

```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>
          <br />
          <br />
        </td>
        <td>
          <div id="clearButton" data-dojo-attach-point="btnClearSelection"
            data-dojo-attach-event="onclick:_onBtnClearSelectionClicked" class="jimu-btn">
            ${nls.buttonTextClearSelection}</div>
          <br />
          <br />
        </td>
      </tr>
    </tbody>
  </table>
</div>
```

Web AppBuilder Widget: JavaScript, Modules

```
1   define([
2     // Specify required modules (dependencies)
3     "dojo/_base/declare",
4     "jimu/BaseWidget",
5     "dijit/_TemplatedMixin",
6     "dijit/_WidgetsInTemplateMixin",
7     "esri/symbols/SimpleFillSymbol",
8     "esri/symbols/SimpleLineSymbol",
9     "esri/tasks/QueryTask",
10    "esri/tasks/query",
11    "esri/toolbars/draw",
12    "esri/Color",
13    "esri/graphic",
14    "dojo/_base/lang",
15    "dojo/_base/array",
16    "dojo/dom",
17    "dojo/dom-style"
18  ],
19    function (
20      // Provide names (aliases) for modules
21      declare,
22      BaseWidget,
23      _TemplatedMixin,
24      _WidgetsInTemplateMixin,
25      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:

```
1  import { React, AllWidgetProps } from "jimu-core";
2
3  function Widget(props: AllWidgetProps<any>) {
4    return <div className="widget-starter jimu-widget">Basic widget...</div>;
5  };
6
7  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

- Styled Components (CSS)

  CSS-in-JavaScript library that bridges the gap between components and styling

  https://styled-components.com

TypeScript: Static Typing

- JavaScript

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

```
1  var x = 3;
2  x = "something";
3  x = true;
```

- TypeScript

```
1  let x: number = 3;
2
3    let x: number
4
5    Type 'string' is not assignable to type 'number'. ts(2322)
6    View Problem (Alt+F8)    No quick fixes available
7  x = "something";
```

Select Features Widget: Imports

```
1   /** @jsx jsx */
2
3   // Import statements.
4   import { React, jsx, css, styled, type AllWidgetProps } from 'jimu-core';
5   import { type JimuMapView, JimuMapViewComponent } from 'jimu-arcgis';
6   import FeatureLayer from 'esri/layers/FeatureLayer';
7   import SimpleLineSymbol from 'esri/symbols/SimpleLineSymbol';
8   import SimpleFillSymbol from 'esri/symbols/SimpleFillSymbol';
9   import GraphicsLayer from 'esri/layers/GraphicsLayer';
10  import FeatureSet from 'esri/rest/support/FeatureSet';
11  import Graphic from 'esri/Graphic';
12  import SketchViewModel from 'esri/widgets/Sketch/SketchViewModel';
13
14  // Widget config object.
15  import { IMConfig } from '../config';
16
17  // Widget messages (strings) object.
18  import defaultMessages from './translations/default';
19
20  // React hooks.
21  const { useState, useEffect } = React;
22
23  function Widget(props: AllWidgetProps<IMConfig>) {
24    // Widget component.
25    const useMapWidgetId: string = props.useMapWidgetIds?.[0];
```

React

ArcGIS Maps SDK for JavaScript 4.x

TypeScript

Select Features Widget: ArcGIS

```
68  function selectFeatures() {
69    // Disable layer popups.
70    featureLayer.popupEnabled = false;
71
72    // Create a new SketchViewModel.
73    const graphicsLayer: GraphicsLayer = new GraphicsLayer();
74    const sketchViewModel: SketchViewModel =
75      new SketchViewModel({
76        layer: graphicsLayer,
77        view: jimuMapView.view,
78        polygonSymbol: selectionSimpleFillSymbol
79      });
80
81    // Listen to the SketchViewModel's create event.
82    sketchViewModel.on('create', (function (event: any) {
83      // Check the event state.
84      if (event.state === 'complete') {
85        // Create a query options object using the user-defined geometry.
86        const queryOptions: IQueryOptions = {
87          where: '',
88          geometry: event.graphic.geometry,
89          returnGeometry: true
90        };
91
92        // Query the features.
```

ArcGIS Maps SDK for JavaScript 4.x

TypeScript

TypeScript

Select Features Widget: Return

```
174   return (
175     // Render the widget UI.
176     <div className='widget-select-features jimu-widget'>
177       <div css={headerStyleLiteral}>{defaultMessages.header}</div>
178       <div>
179         <StyledButton
180           onClick={handleSelectFeaturesButtonClick}
181           disabled={!configured}>
182           {defaultMessages.btnSelectFeaturesCaption}
183         </StyledButton>
184          
185         <StyledButton
186           onClick={handleClearSelectionButtonClick}
187           disabled={!configured}>
188           {defaultMessages.btnClearSelectionCaption}
189         </StyledButton>
190       </div>
191       {
192         useMapWidgetId &&
193         <JimuMapViewComponent useMapWidgetId={useMapWidgetId}
194           onActiveViewChange={setJimuMapView} />
195       }
196     </div>
197   )
198 }
```

JSX

Emotion variable

Styled Components

Select Features Widget: Styles

```
155  // Create a style literal for the header.
156  const headerStyleLiteral: string = css`
157      color: ${props.theme.colors.black};
158      font-size: 1rem;
159      font-weight: bold;
160    `;
161
162  // Create a styled button.
163  const StyledButton: any = styled.button`
164      ${configured && css`
165        color: white;
166        background-color: ${props.theme.colors.primary};
167      `}
168      ${!configured && css`
169        color: white;
170        background-color: ${props.theme.colors.secondary};
171      `}
172    `;
173
174  return (
175    // Render the widget UI.
176    <div className='widget-select-features jimu-widget'>
177      <div css={headerStyleLiteral}>{defaultMessages.header}</div>
178      <div>
179        <StyledButton
```

Emotion variable

Styled Components

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

Counties

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

Table Widget Style Configuration

- Set column colors according to configuration values

Field Name	Column Color
STATEFP	orange
COUNTYFP	aqua
COUNTYNS	gray

- Allow for row selection

Table Widget Styling

Counties

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

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

```
// All material copyright ESRI, All Rights Reserved, unless otherwise
specified.
// See https://js.arcgis.com/4.27/esri/copyright.txt for details.
//>>built
define("require ../chunks/_rollupPluginBabelHelpers ../chunks/tslib.es6
../intl ../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 ../intl/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,menuItems:{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()),r.on(()=>this.highlightIds,"change",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
  https://developer.mozilla.org/en-US/docs/Web/API/Web_Components

- The web component includes a template grid consisting of placeholders called "slots" that can be filled with markup
  https://developer.mozilla.org/en-US/docs/Web/HTML/Element/Slot

- 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

```
<vaadin-grid-cell-content slot="vaadin-grid-cell-con
tent-88" title="2">2</vaadin-grid-cell-content>
slot == $0
```

Slot Content

```
▼<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: 2
 0000000;"> flex
 </td>
```

Selected Cell - aria-selected

```
▼<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: 2
0000000;"> 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."

https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA

Column Cell Style Class

Create an HTML style class

- E.g.,

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

<class name><selector----------><rule set---------->

- 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

Counties

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

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
  https://www.esri.com/arcgis-blog/products/web-appbuilder/announcements/arcgis-web-appbuilder-roadmap-for-retirement

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

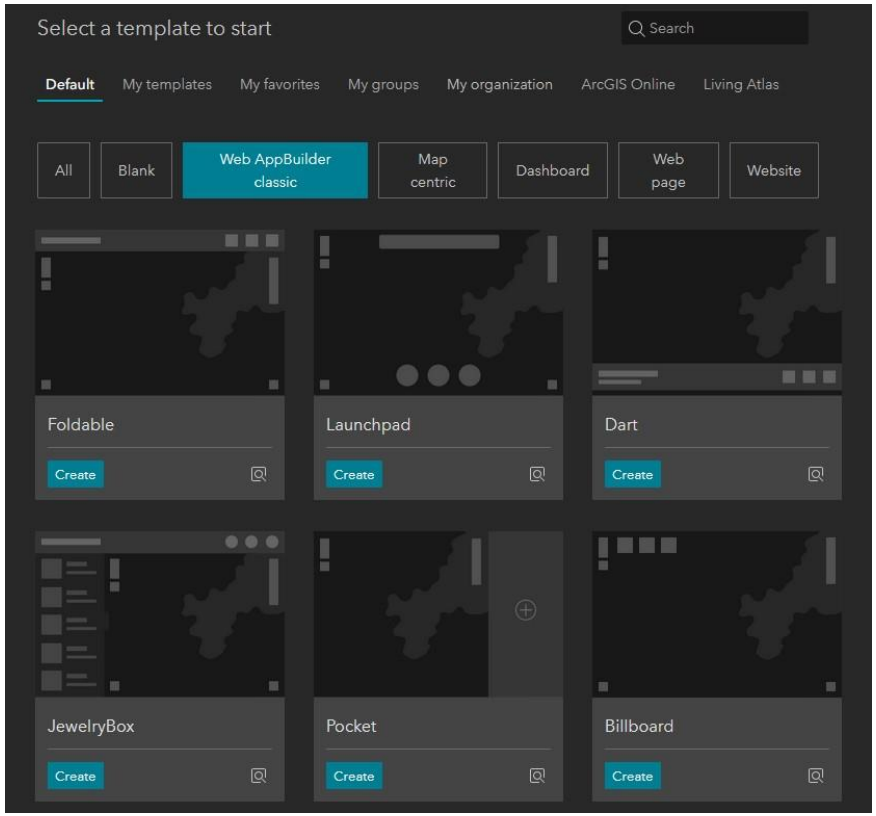
- Experience Builder has reached parity with Web AppBuilder, but the functional correspondence is not necessarily one-to-one
  It may be tricky to find what you need

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



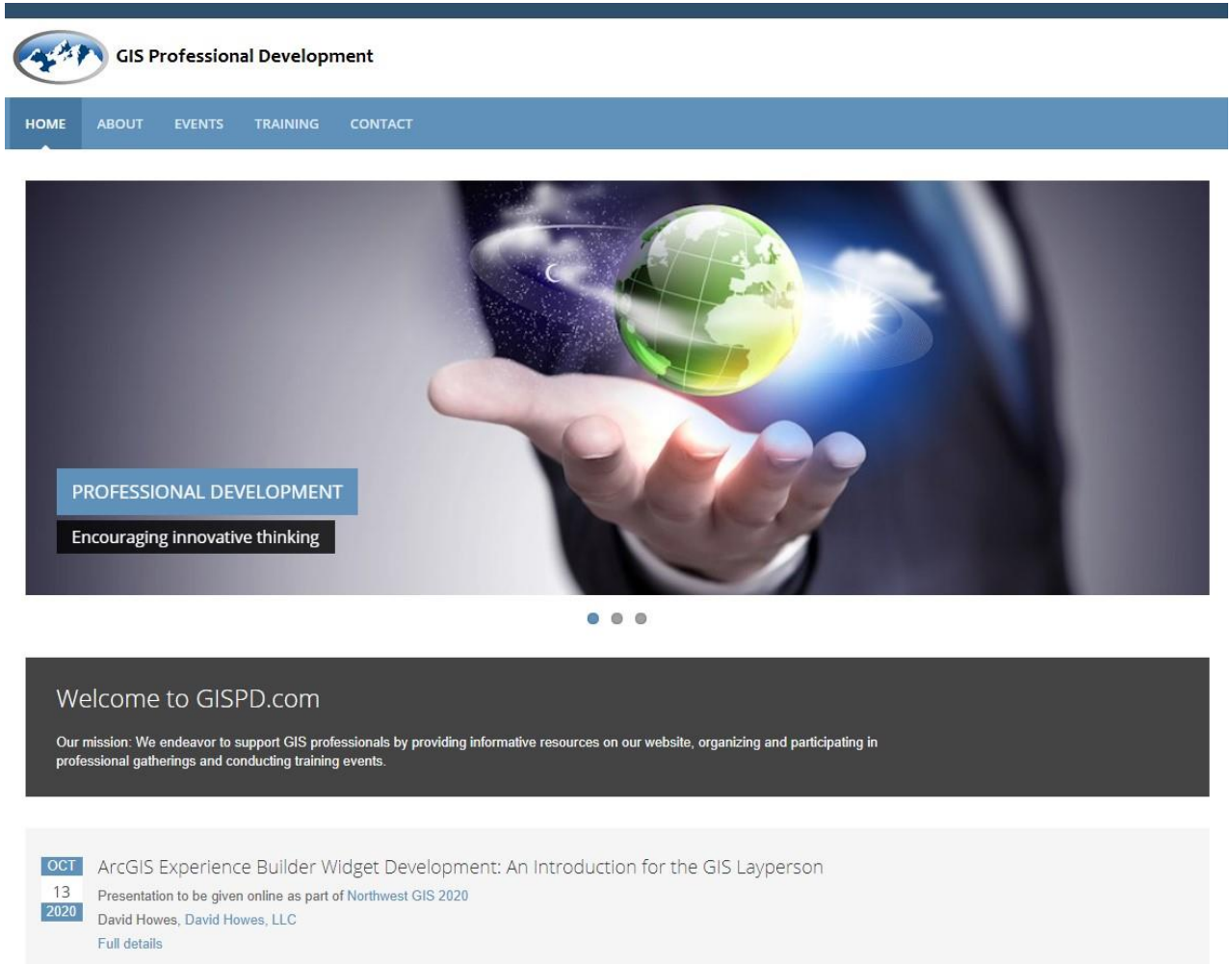
Discussion

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