



ArcGIS API for JavaScript: What's New

Julie Powell | Noah Sager



A decorative background featuring a topographic map with contour lines. Overlaid on the map are several large, semi-transparent geometric icons in cyan, yellow, and orange, including crosses, squares, and L-shapes. In the bottom right corner, there is a vertical text block with the words "SEE WHAT OTHERS CAN'T" in a light blue font. The overall aesthetic is modern and professional, suggesting advanced technology and data analysis.

SEE
WHAT
OTHERS
CAN'T

Welcome!



Release notes for 4.12

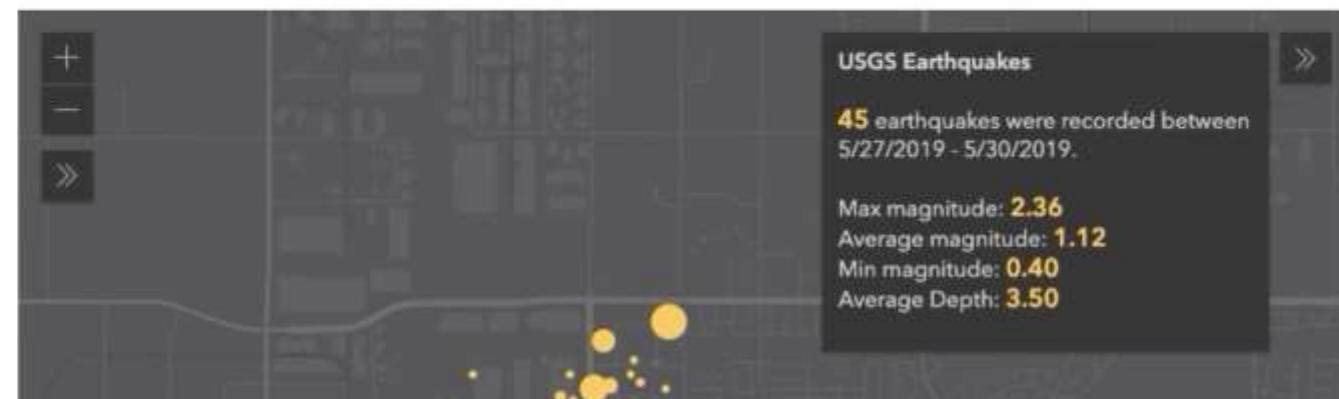
Overview
Release notes
Get the API
Quick Start
> Tutorials
> Core Concepts
> Data Visualization
> Building your UI
> Working with ArcGIS Online and Enterprise
> Developer Tooling
> Migrating from 3.x
> Reference

Time

We increased our capabilities to visualize temporal data in both [2D MapViews](#) and [3D SceneViews](#). We added the [TimeInterval](#) class to describe a length of time in different temporal units, which is referenced by time-aware layers and the [TimeSlider](#) widget.

The beta version of the [TimeSlider](#) widget simplifies time manipulation in your application, and can be configured to update the View's [timeExtent](#), which means all time aware layers will update their contents to conform to this change. You can also use [TimeSlider](#) widget to visualize temporal data on the client-side by setting filters or effects on [FeatureLayerView](#), [CSVLayerView](#), and [GeoJSONLayerView](#).

We will continue to add more support for time-awareness. This includes, but is not limited to, adding [timeExtent](#) properties on layers that store temporal data, allowing layers to follow their own timeline without having to follow [View.timeExtent](#), and continued improvements and enhancements to the [TimeSlider](#) widget.



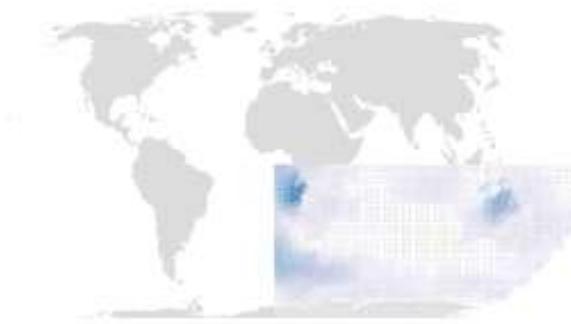
Content
Time
Performance improvements
API Modernization
Client-side queries in 3D
Water rendering
2D WebStyleSymbols
New 3D WebStyleSymbols
Smart Mapping updates
Scale-dependent visualizations
Dot density
3D support for lines and polygons
Slider widgets
Color scheme updates
New 3D Line Symbols
Filters on BuildingSceneLayer
Asynchronous Method Cancellation
Geodetic computations
Labeling updates
Data and number formatting

PERFORMANCE

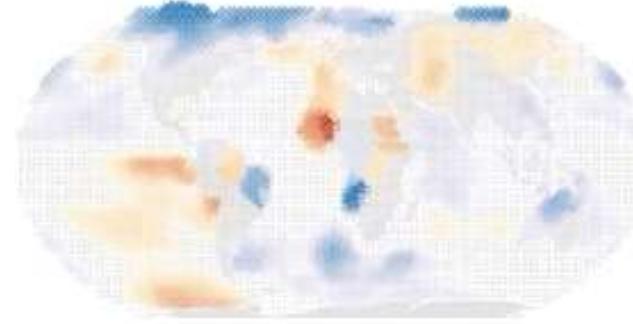
Faster loading. >100kb less JS
Vector tile optimizations
Faster rendering of line features
Highly performant FeatureLayers through Feature Tiles
Fast renderer updates → no flashing!

4

4.11



4.12



1900

4.11

4.12

FEATURE [TILE] LAYERS

Maximizing performance: a look under the hood

1. Query in an efficient way
2. Minimize size of data delivered to browser
3. Fast rendering



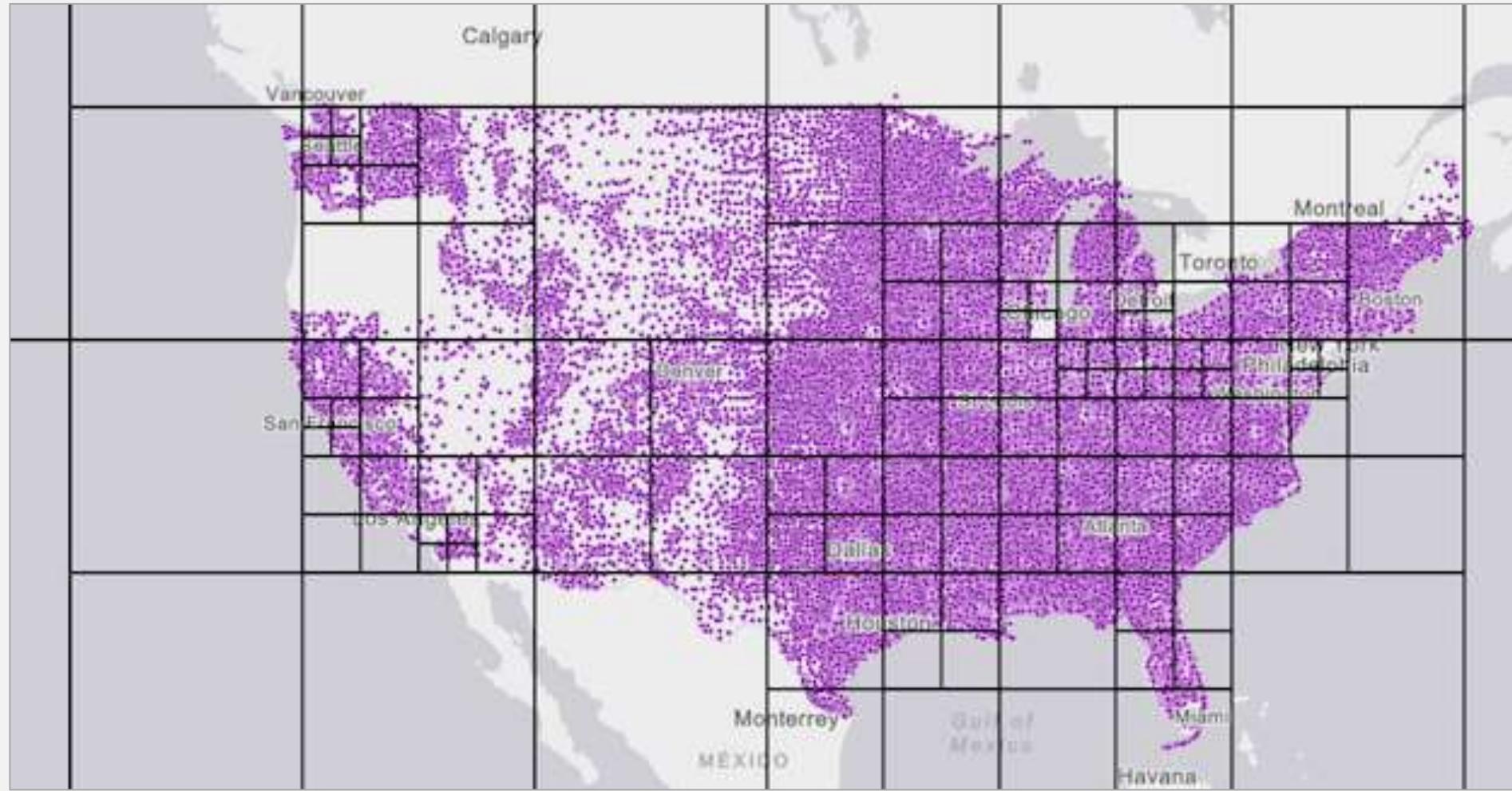
FEATURE [TILE] LAYERS

+

Maximizing performance: a look under the hood

1. Query in an efficient way -> feature tiles & caching
2. Minimize size of data delivered to browser-> *binary format (PBF)* & brotli compression
3. Fast rendering -> WebGL (all layers)





FEATURE FETCH STRATEGY

- Feature tile queries
- Progressive feature tile subdivisions
- Smaller tiles in feature dense areas

query?f=json... h2 720 ms 51 ms gzip Hit from cloudfront

query?f=pbf... h2 8.0 KB 154 ms gzip RefreshHit from cloudfront

query?f=pbf... h2 15.9 KB 35 ms gzip Hit from cloudfront

query?f=pbf... h2 367 KB 1.57 s gzip Hit from cloudfront

query?f=pbf... h2 66.1 KB 119 ms gzip Hit from cloudfront

query?f=pbf... h2 1.1 KB 124 ms gzip Hit from cloudfront

query?f=pbf... h2 79.2 KB 133 ms gzip Hit from cloudfront

query?f=pbf... h2 10.6 KB 131 ms gzip Hit from cloudfront

query?f=pbf... h2 156 KB 119 ms gzip Hit from cloudfront

query?f=pbf... h2 29.1 KB 25 ms gzip Hit from cloudfront

query?f=pbf... h2 72.8 KB 23 ms gzip Hit from cloudfront

query?f=pbf... h2 37.9 KB 30 ms gzip Hit from cloudfront

query?f=pbf... h2 32.9 KB 21 ms gzip Hit from cloudfront

query?f=pbf... h2 16.8 KB 52 ms gzip Hit from cloudfront

query?f=pbf... h2 48.5 KB 159 ms gzip RefreshHit from cloudfront

ms gzip Hit from cloudfront

ms gzip RefreshHit from cloudfront

ms gzip Hit from cloudfront

ms gzip RefreshHit from cloudfront

ms gzip Hit from cloudfront

ms gzip RefreshHit from cloudfront

ms gzip Hit from cloudfront

ms gzip Hit from cloudfront

ms gzip Hit from cloudfront

ms gzip RefreshHit from cloudfront

ms gzip Hit from cloudfront

ms gzip RefreshHit from cloudfront

ms gzip Hit from cloudfront

HIGH PERFORMANCE FEATURE LAYERS

Efficient querying with feature tiles & caching
PBF
WebGL

PERFORMANCE

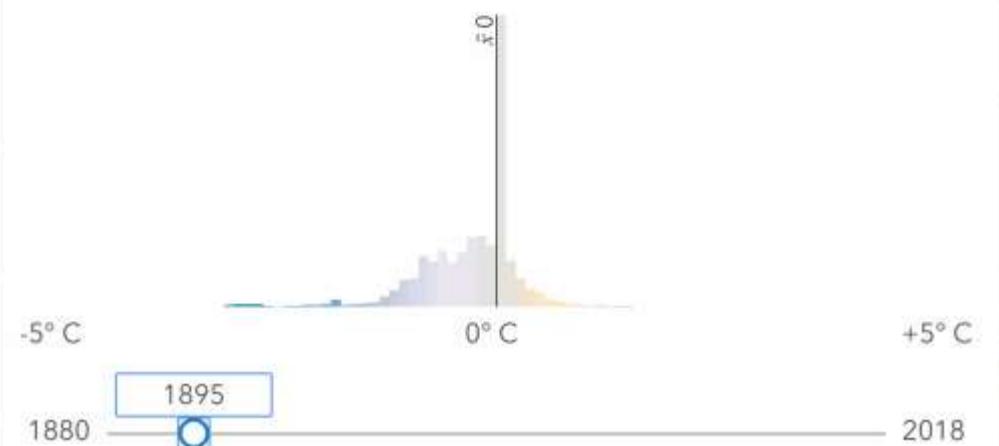
Improved integrated mesh performance
Fast feature layers

9



Visualization

Temperature Anomaly





VECTOR TILE LAYERS

Style esri's basemaps
Or create your own



BETA

Newspaper << Land/pattern <<

Search Layers

Visibility

Visible 2

Visible Zoom Range 0 World Buildings 18

Appearance

Color No Color

Outline Color No Color

Pattern Edit image in all layers
City small scale/medium other capital

Opacity 24 %

Position

Translate x 0 y 0

Esri, FAO, NOAA

Click on any map to edit that layer

Find address or place

Land

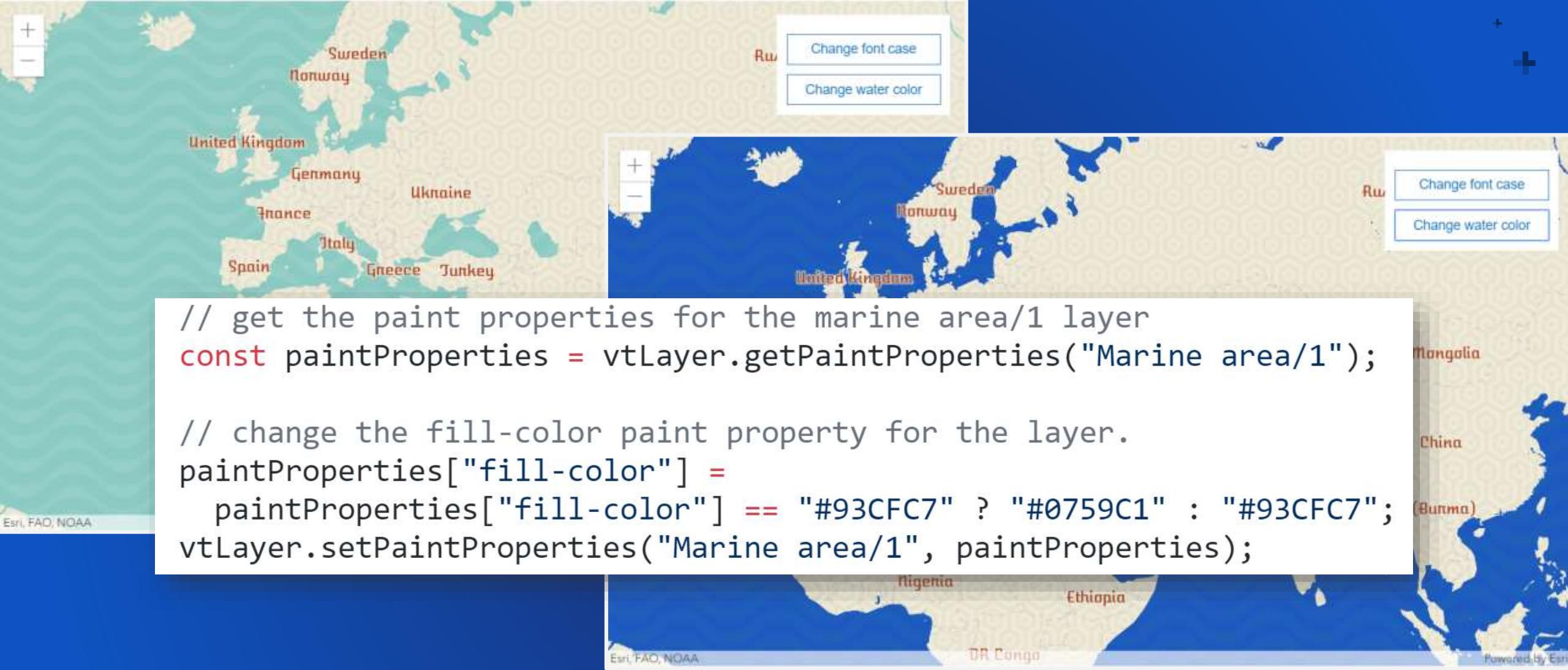
- Land
- Land/pattern
- Forest Or Park
- Special Area Of Interest
- Populated Places
- Land Use
- Transportation
- Boundaries
- Water
- Buildings
- Roads

Powered by Esri

No Background | Hillshade | Imagery | Hide Minimaps

VECTOR TILE LAYERS

Style the map ahead of time with the Vector Tile Style Editor



VECTOR TILE LAYERS

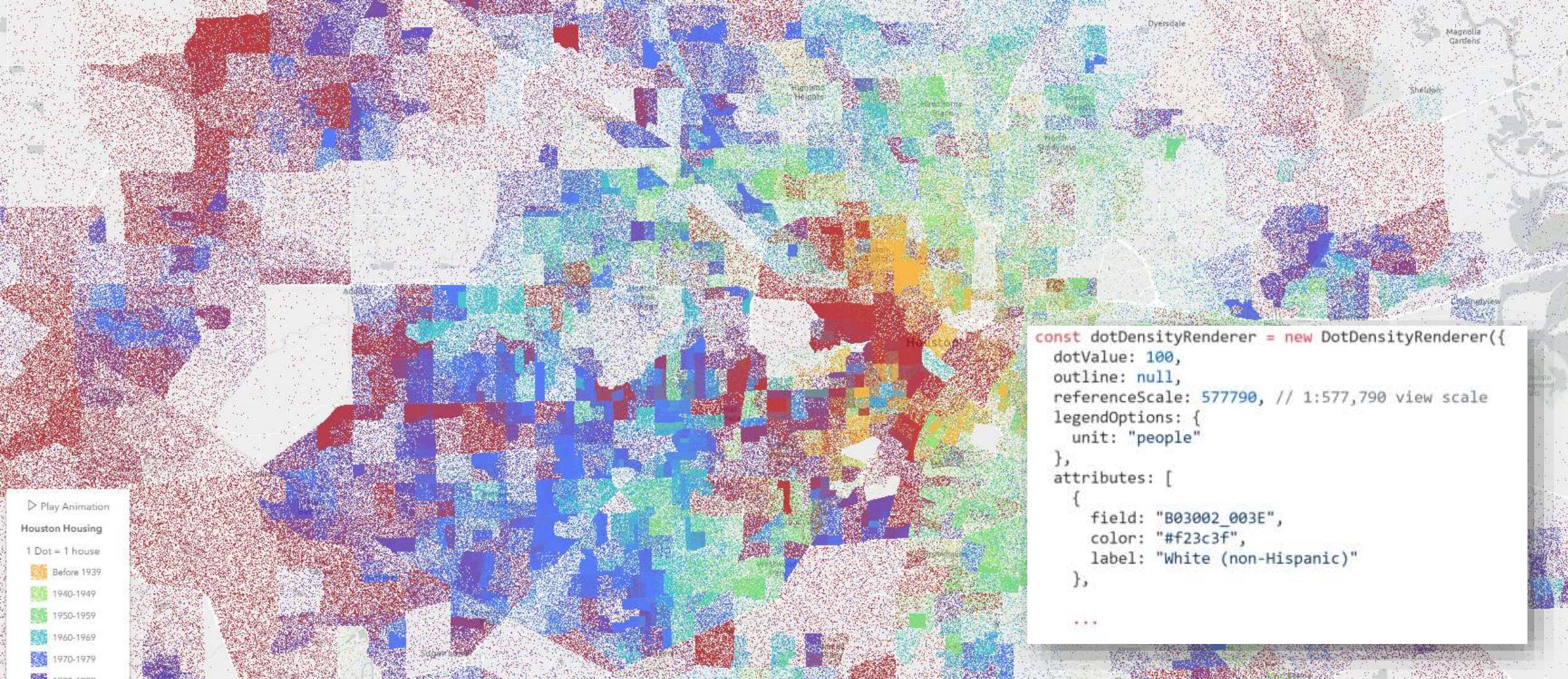
Load the style of your choice, or
Style the layer in code.
Option to update it without a reload (4.10)



DATA-DRIVEN STYLING

Where?
What?
When?
How much?

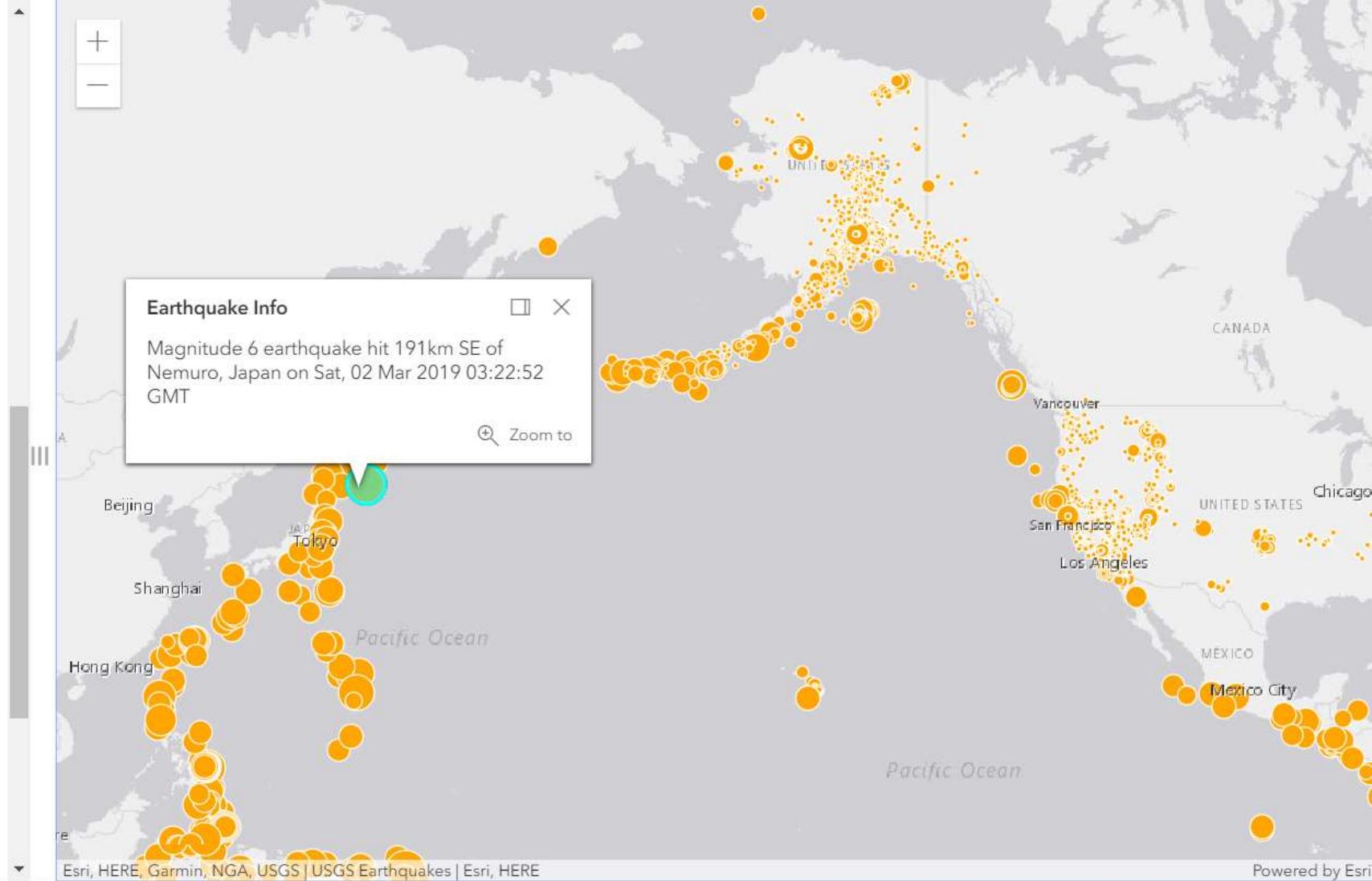




DOT DENSITY

Randomly drawn dots to represent a field value.
Configure how much each dot represents

```
47
48
49 const template = {
49   title: "Earthquake Info",
50   content: "Magnitude {mag} {type} hit {place} on
50   {time:DateString}"
51 };
52
53 const renderer = {
54   type: "simple",
55   field: "mag",
56   symbol: {
57     type: "simple-marker",
58     color: "orange",
59     outline: {
60       color: "white"
61     }
62 },
63 visualVariables: [
64   {
65     type: "size",
66     field: "mag",
67     stops: [
68       {
69         value: 2.5,
70         size: "4px"
71       },
72       {
73         value: 8,
74         size: "40px"
75       }
76     ]
77   }
78 ];
79
80 const geojsonLayer = new GeoJSONLayer({
81   url: url,
82   copyright: "USGS Earthquakes",
83   popupTemplate: template,
84   renderer: renderer //optional
85 });
86
87 const map = new Map({
```



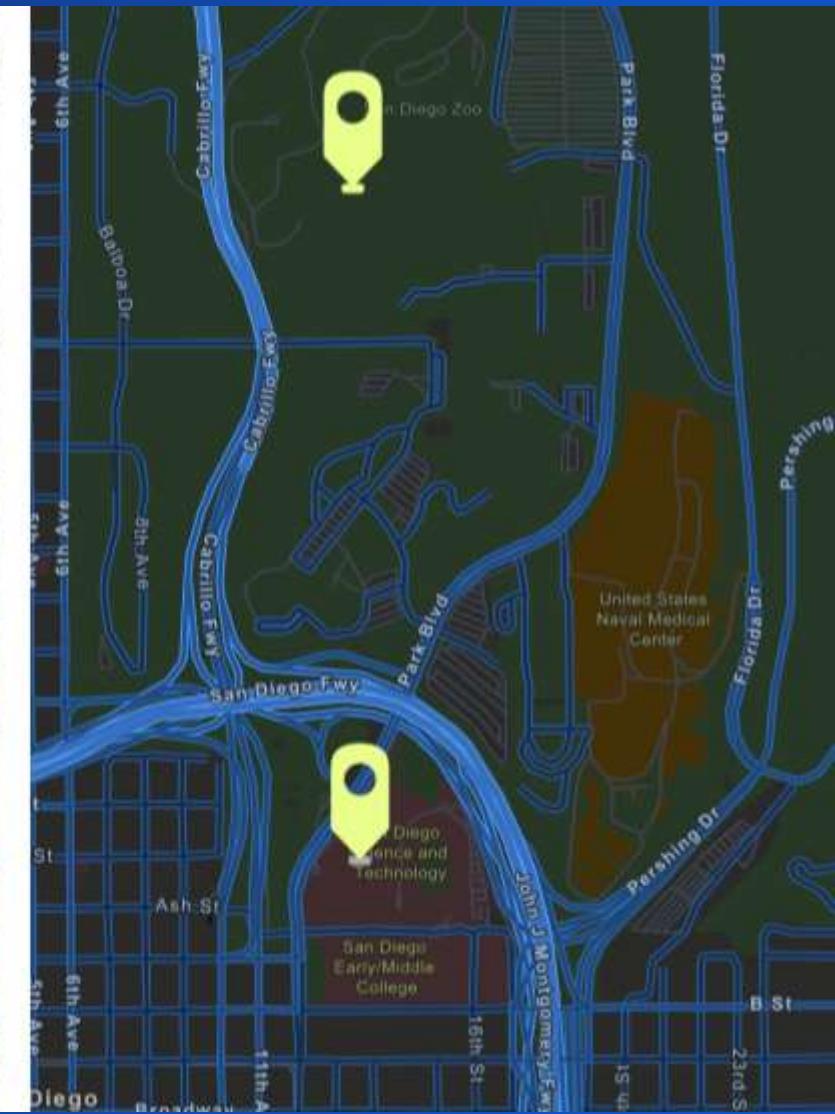
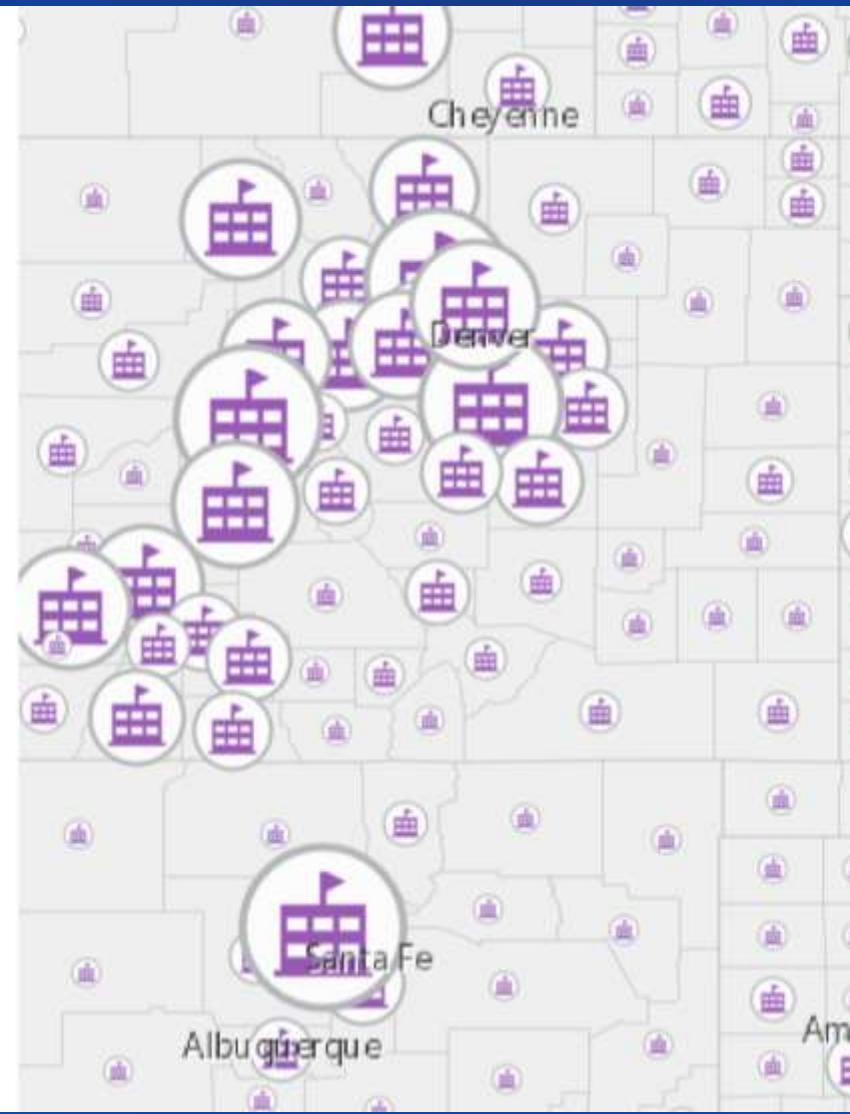
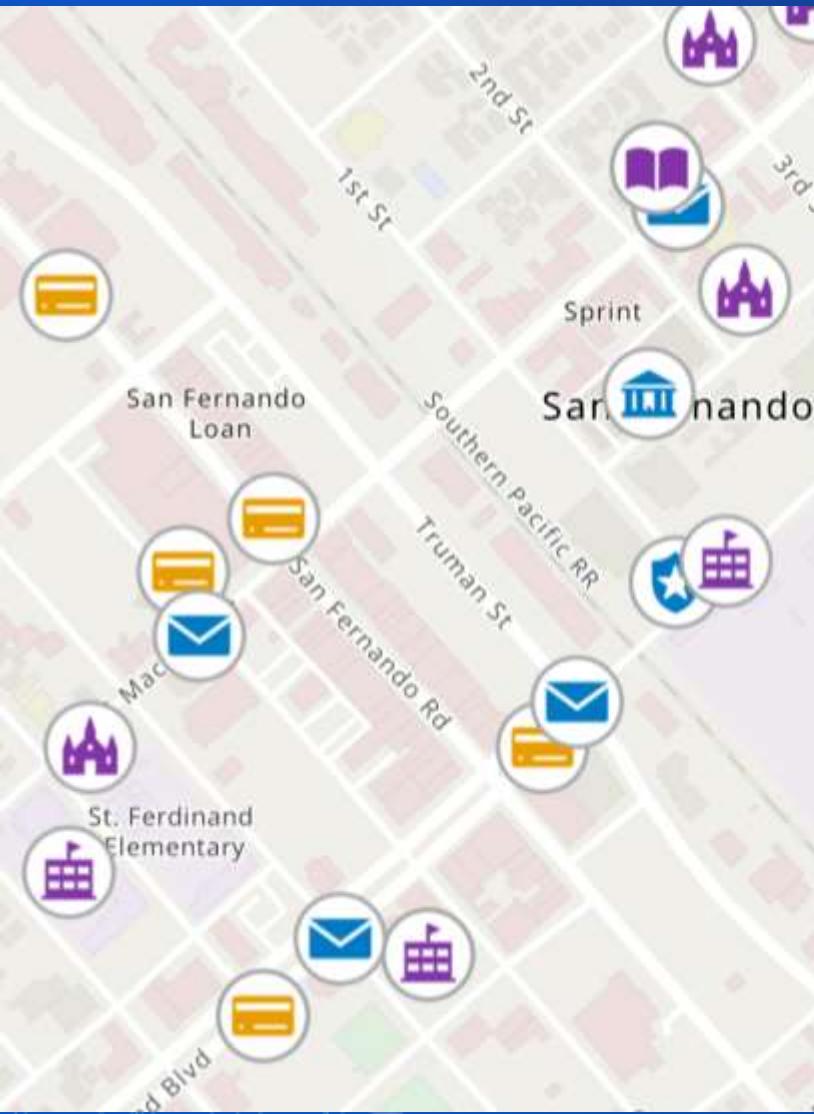
GEOJSON LAYER

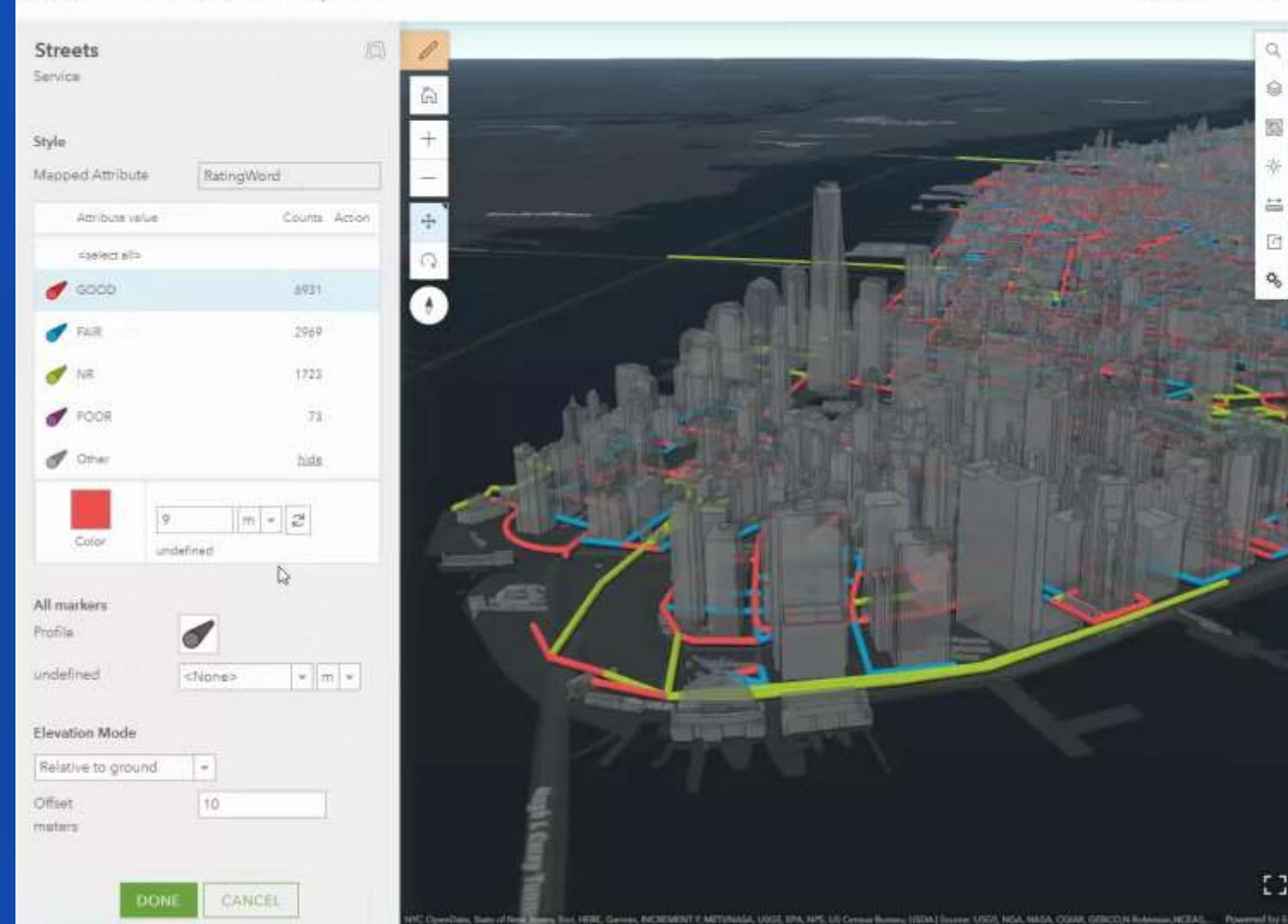
Style & interact like a feature layer

VECTOR MARKER SYMBOLS

More than 100 new 2D web style symbols
CIM symbols

19



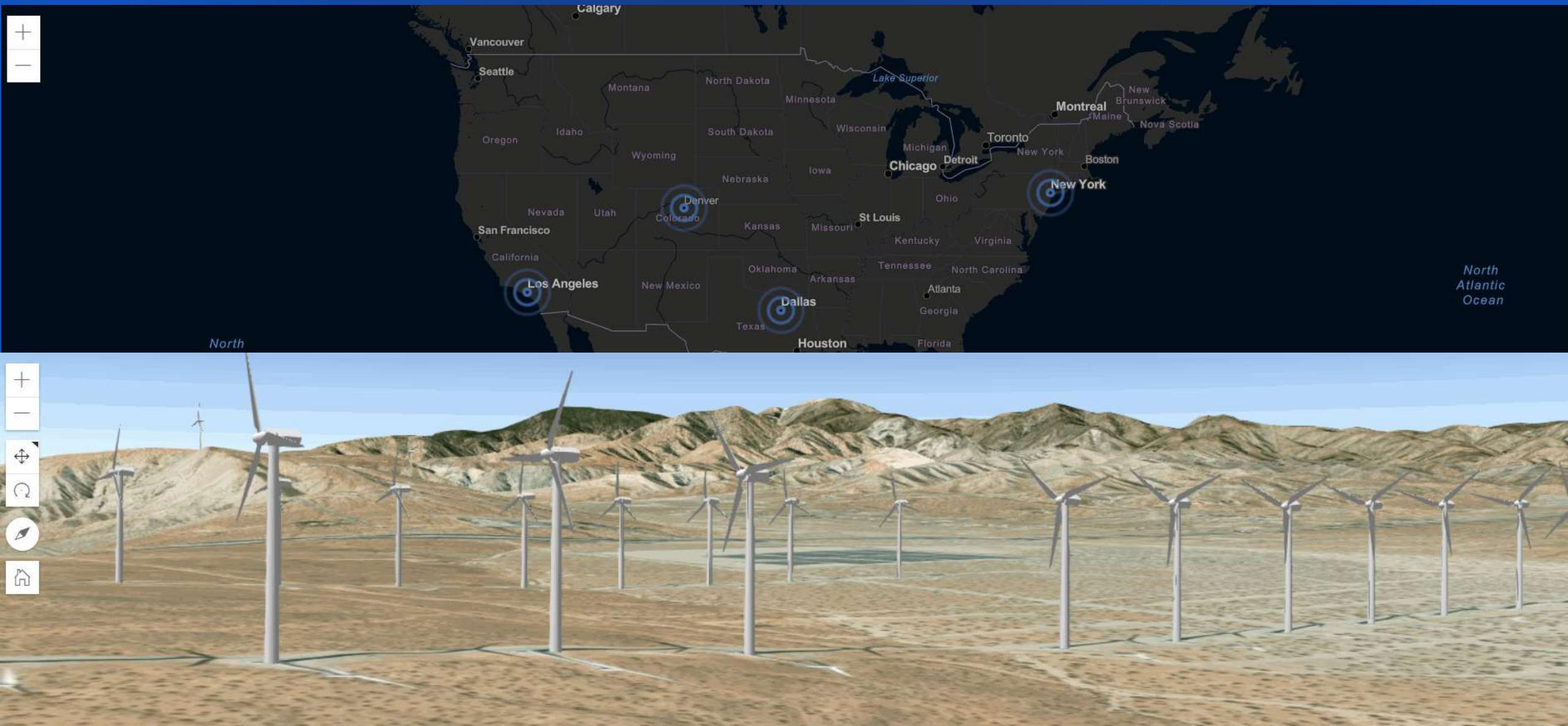


SMART MAPPING

Parity with 3.x, plus more. i.e.:

- Scale-driven outline thickness
- Icon sizing based on scale

Create custom WebGL layer views

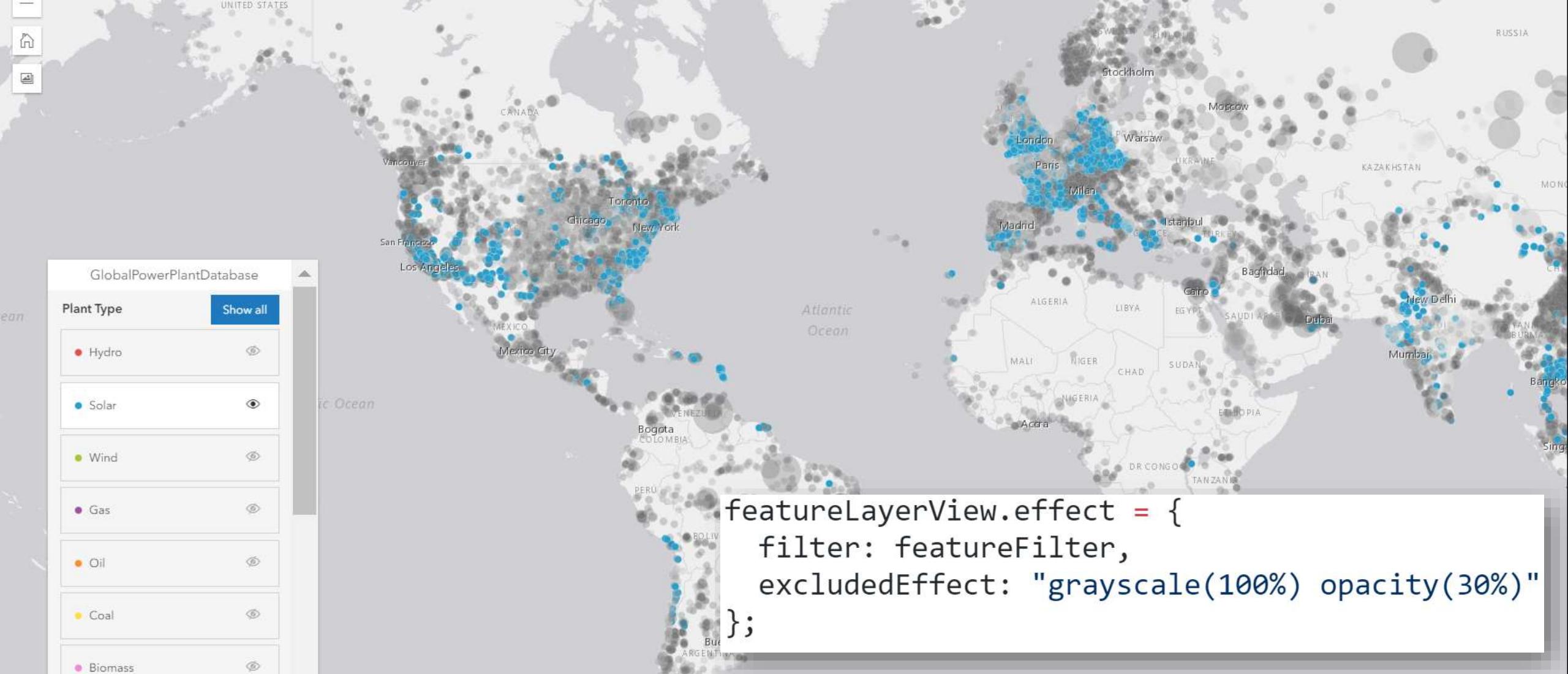


INTERACTIVITY

Client-side

- Querying
- Filtering
- Statistics
- Geometric operations





```
featureLayerView.effect = {  
    filter: featureFilter,  
    excludedEffect: "grayscale(100%) opacity(30%)"  
};
```

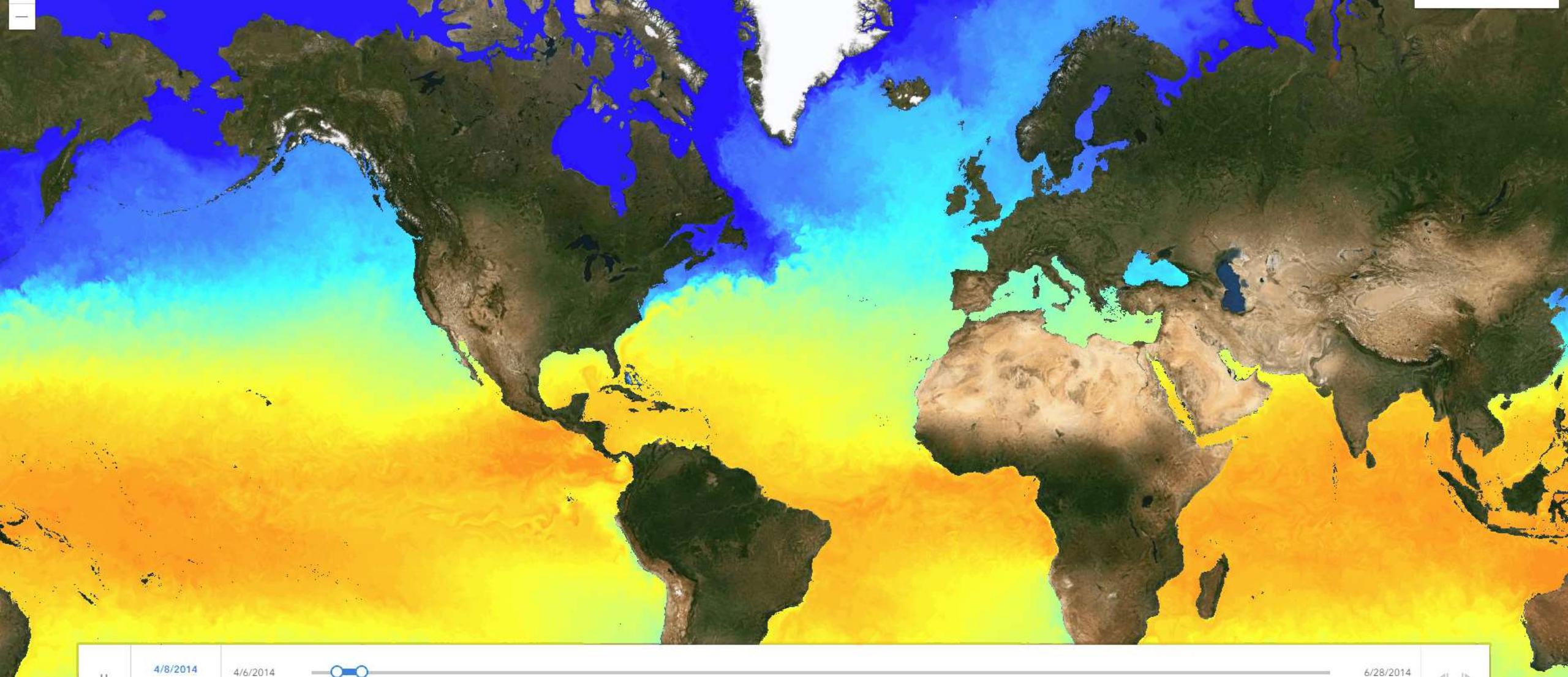
FILTERING

Client-side

Decide how to style feature within filter, and outside filter.

Time





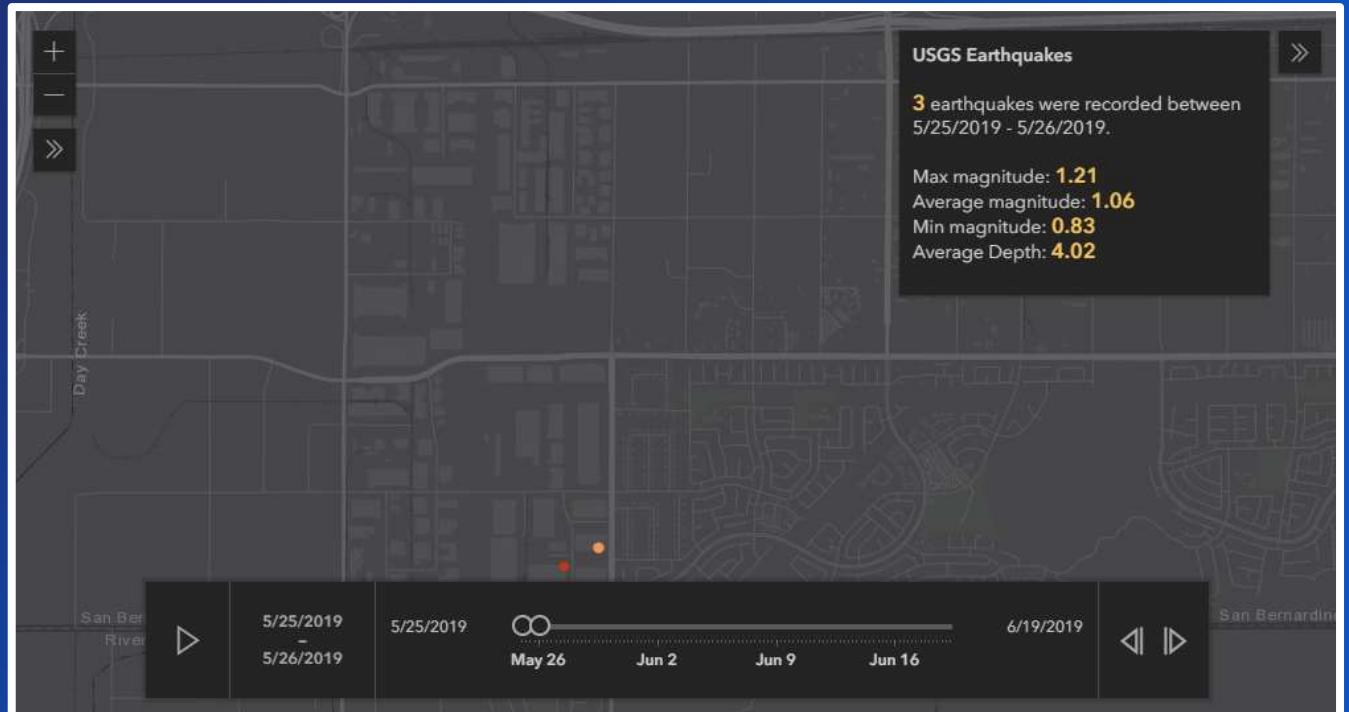
TIME

Time aware layers and views
Time slider widget

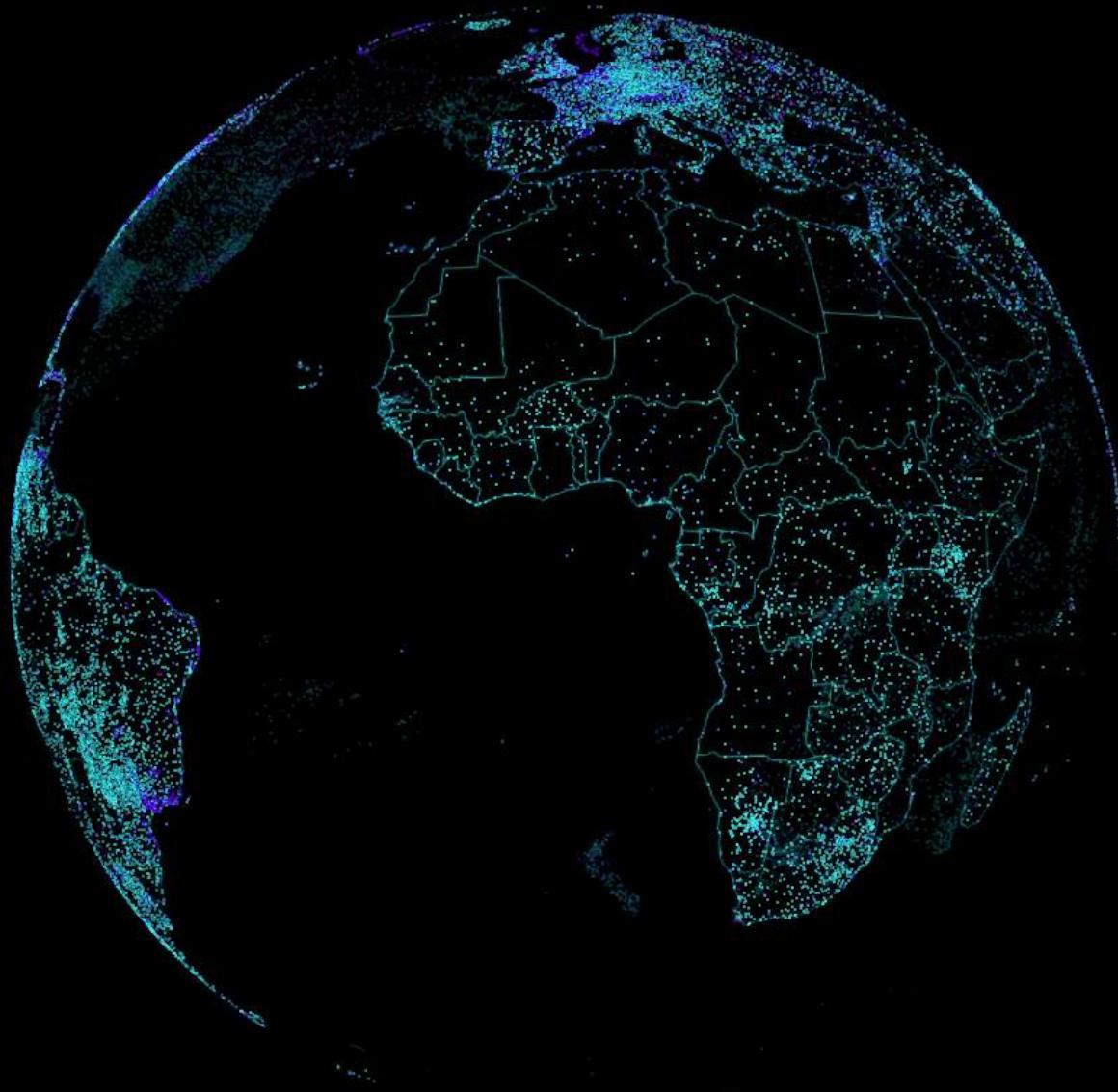


TIME

1. **timeInfo** on the layer
2. **timeExtent** on the view
3. **TimeSlider** widget

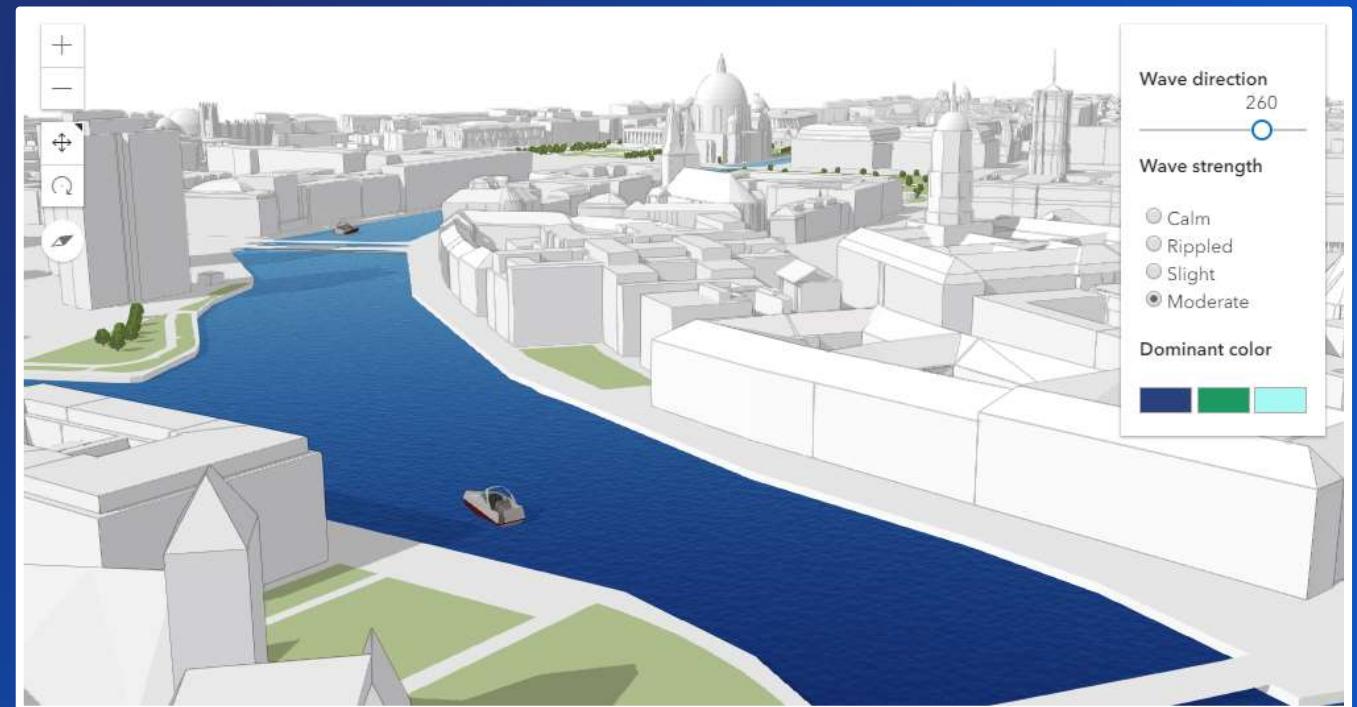


3D



REALISTIC RENDERING

1. WaterSymbol3DLayer
2. Esri Web Style Symbols
3. glTF models



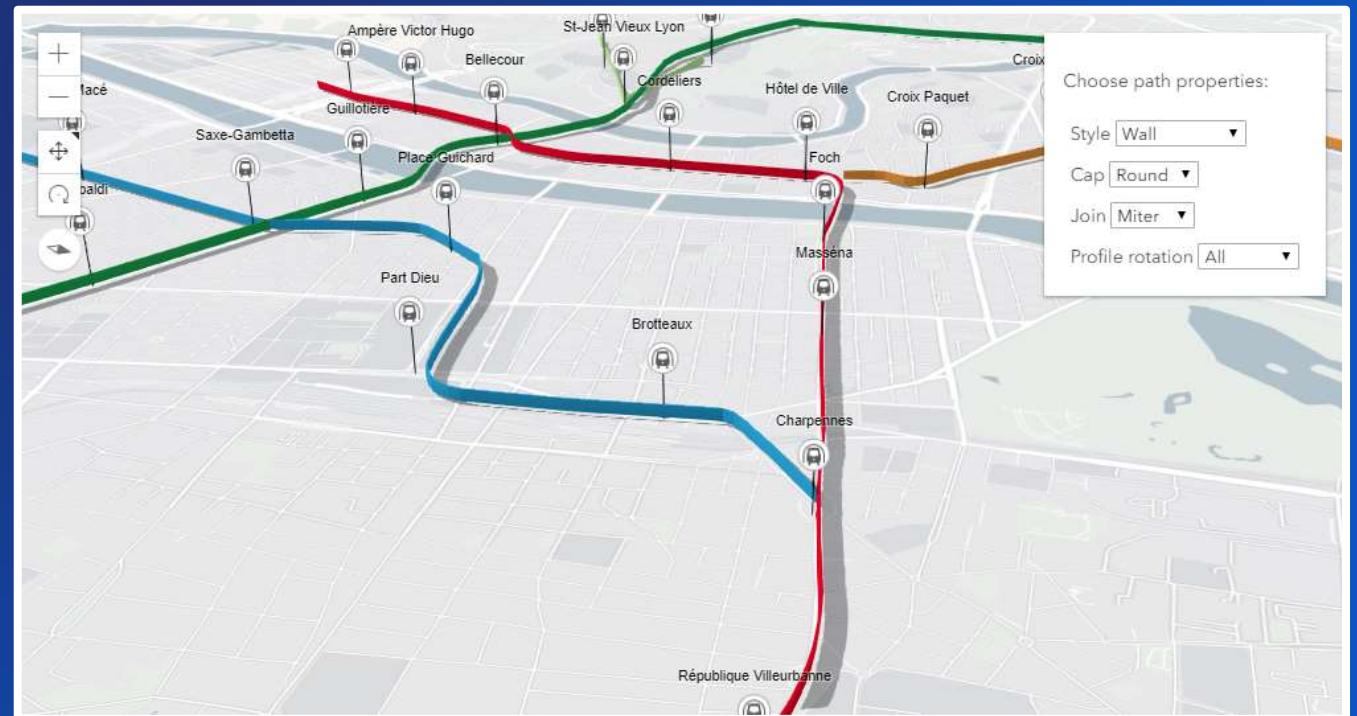
BUILDING SCENE LAYER

1. Detailed exteriors / interiors
2. Slice widget
3. Filtering

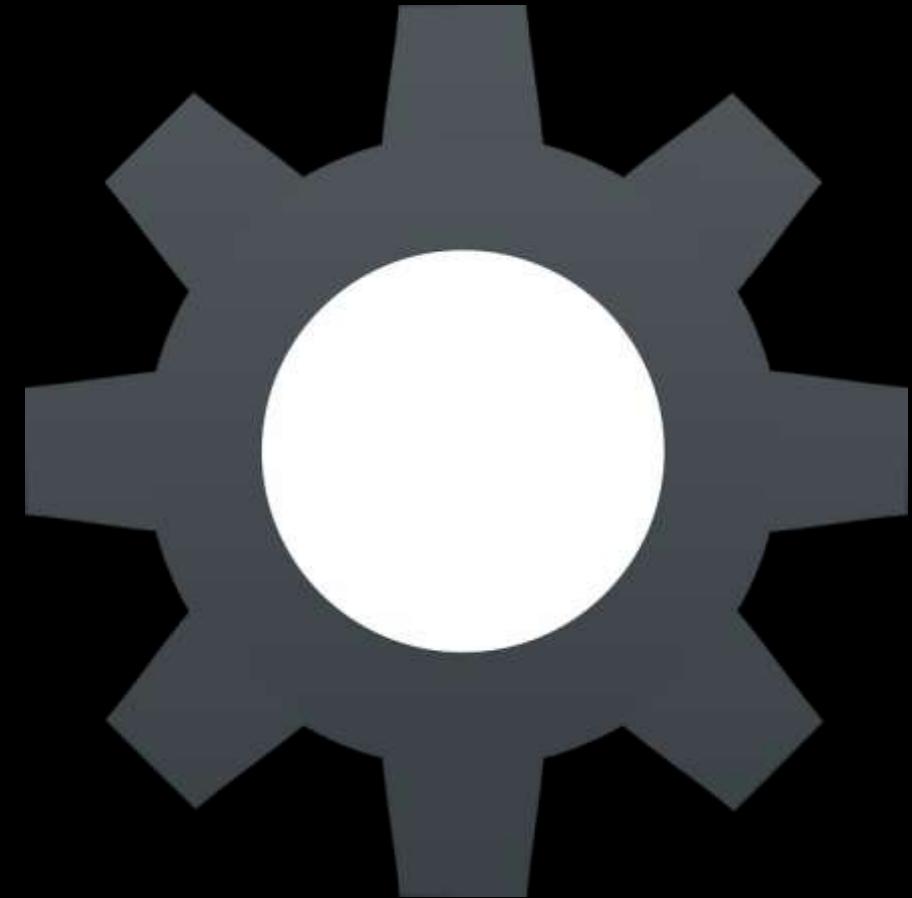


LINE SYMBOLS

1. Round tube
2. Square tube
3. Wall
4. Strip



Widgets



Widgets

BasemapGallery widget

Bookmarks widget

CoordinateConversion widget

CoordinateConversion widget - custom formats

Directions widget

LayerList widget

LayerList widget with actions

Legend widget

Add a Legend to LayerList

Legend widget card style

Locate button

Measurement in 2D

Measurement in 3D

BuildingSceneLayer with Slice widget

Print widget

Track current location

Track widget simulation

Expand widget

Feature widget

Using the view's UI

Responsive widgets

Responsive apps using CSS

TimeSlider Widget

Widgets (Advanced)

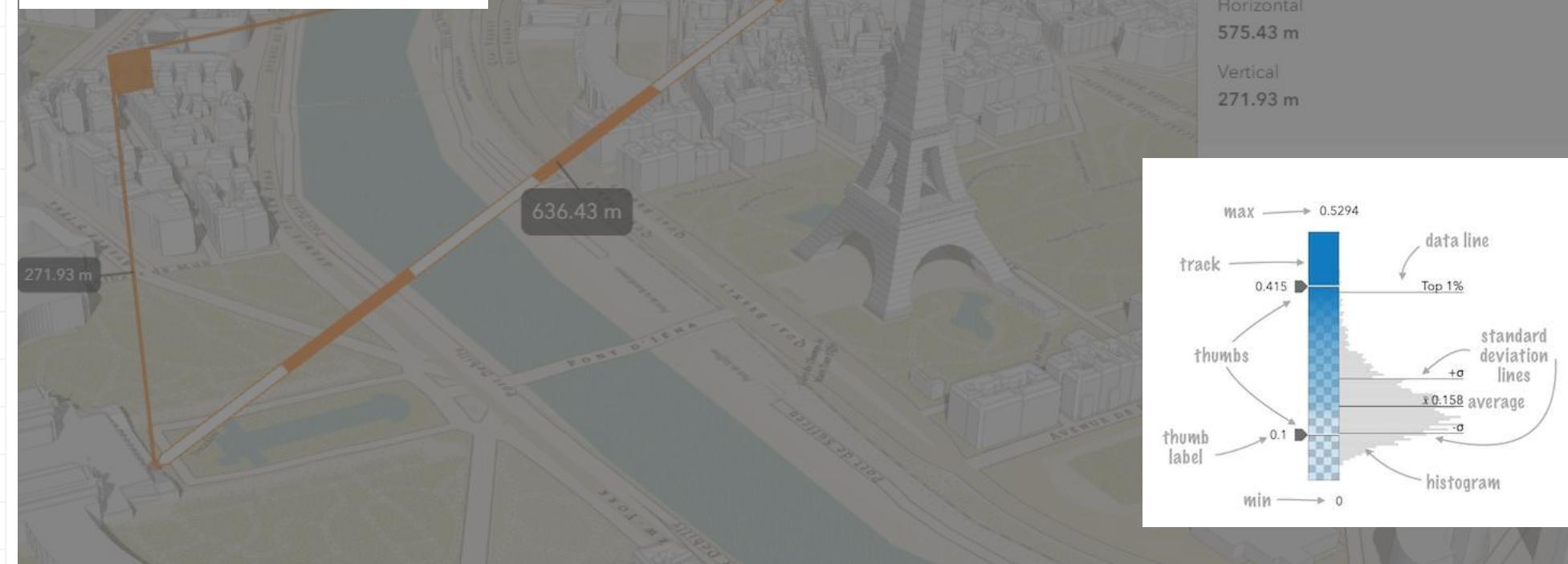
Create a custom widget

Custom Recenter widget

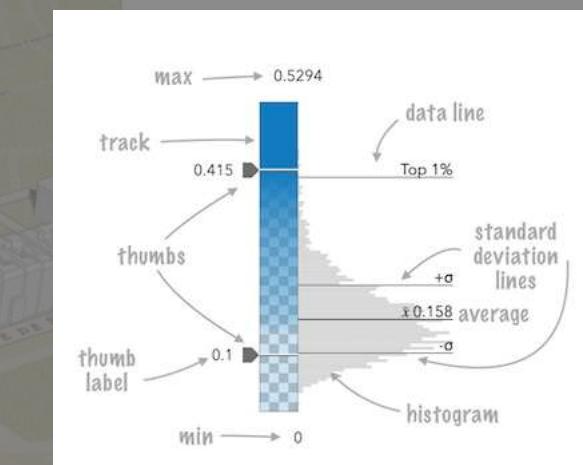
Using widgets with React

Using widgets with Riot

Custom widgets with Vue

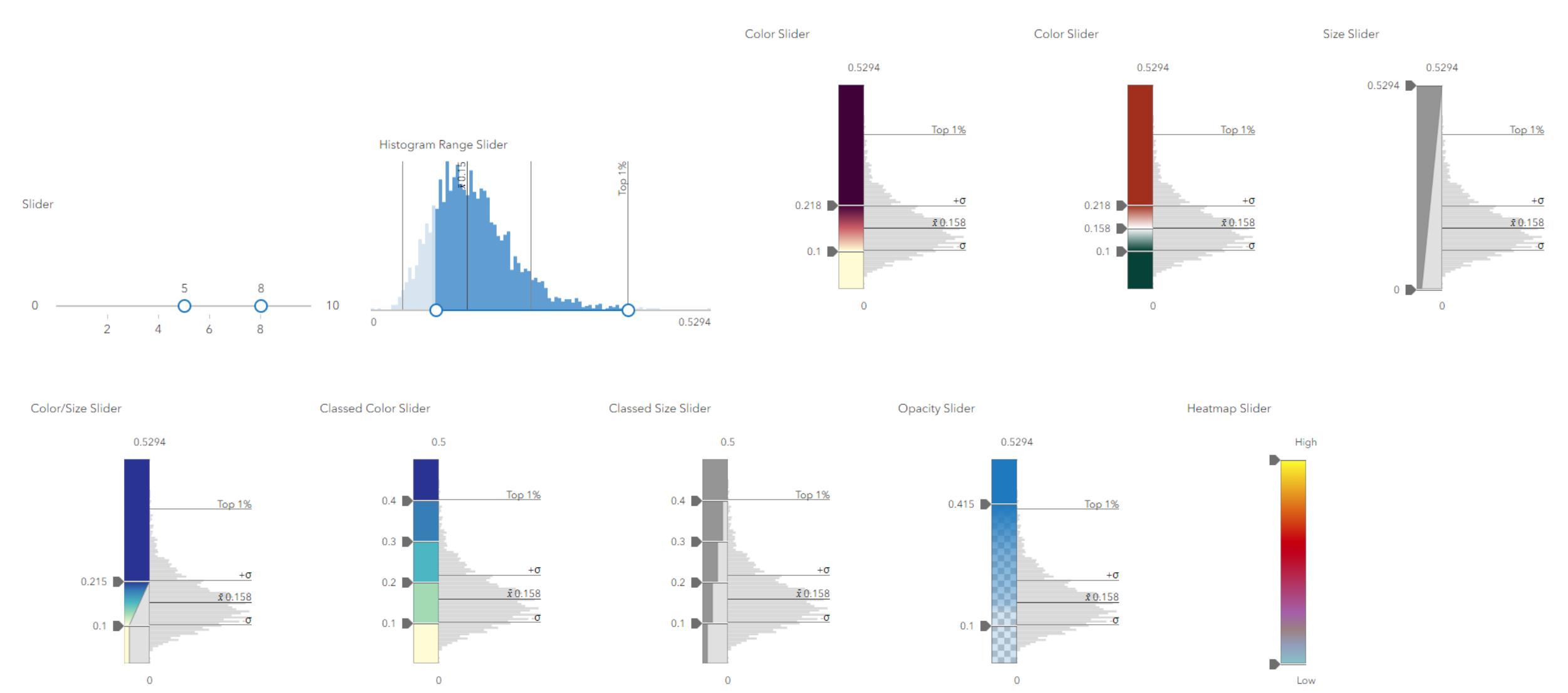


```
var editor = new Editor({  
  view: view  
});  
  
view.ui.add(editor, "top-right");
```



BUILD YOUR UI

Collection of widgets
Customizable
Easy placement

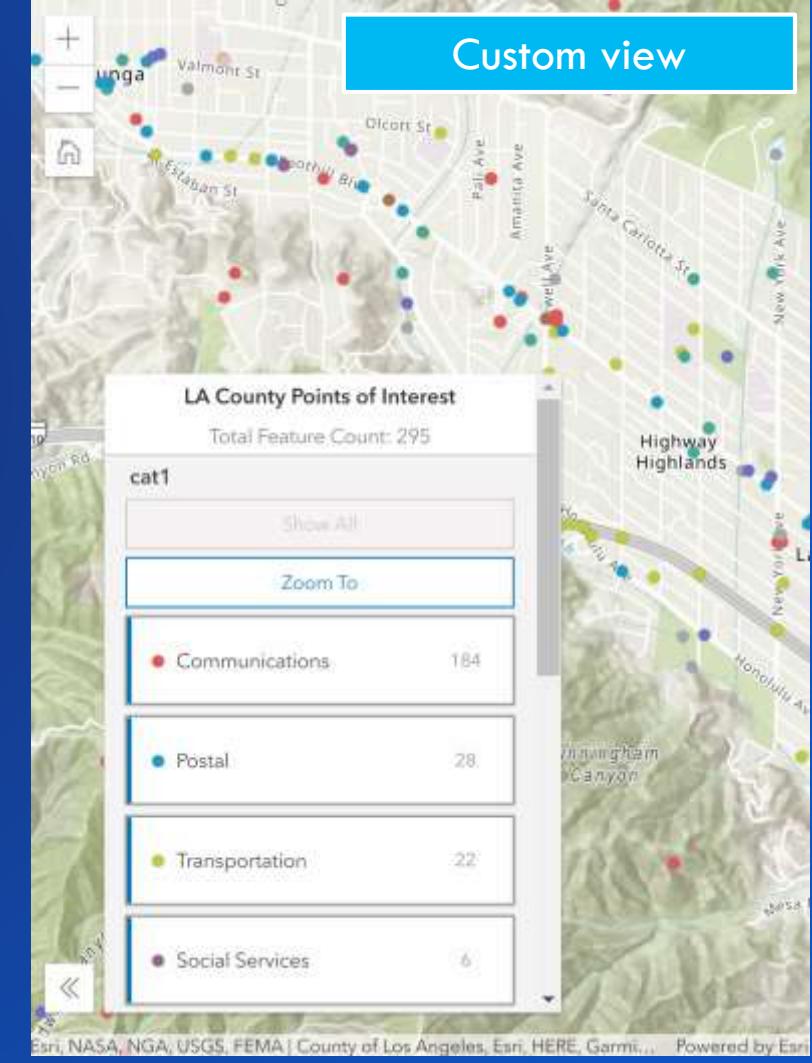
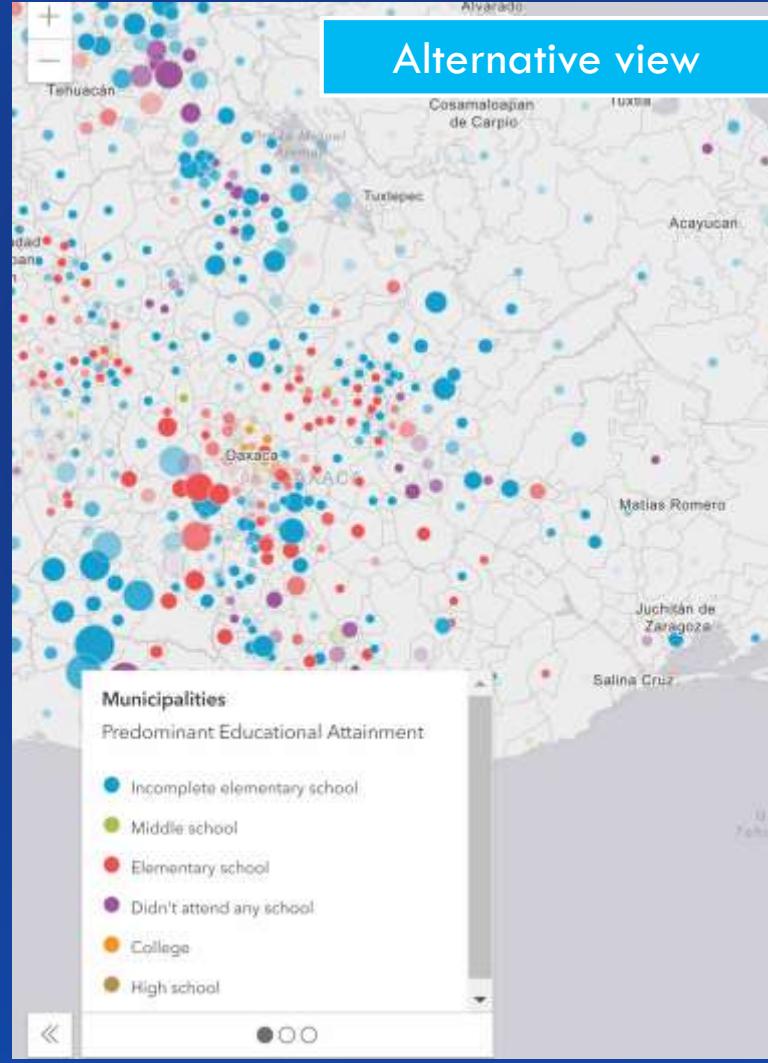
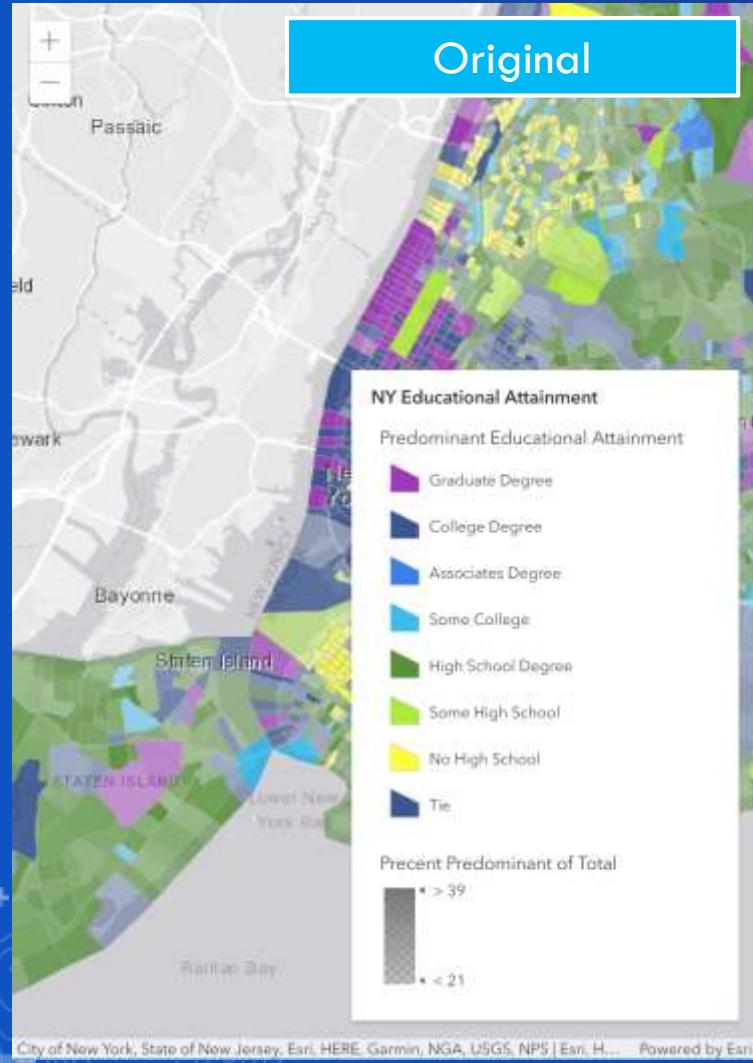


SLIDERS

- Generic slider class
- Useful for any kind of numeric or range of values

WIDGETS

- Use OOB or customize/extend
- Widget view / view model architecture
- On deck: Layer swipe



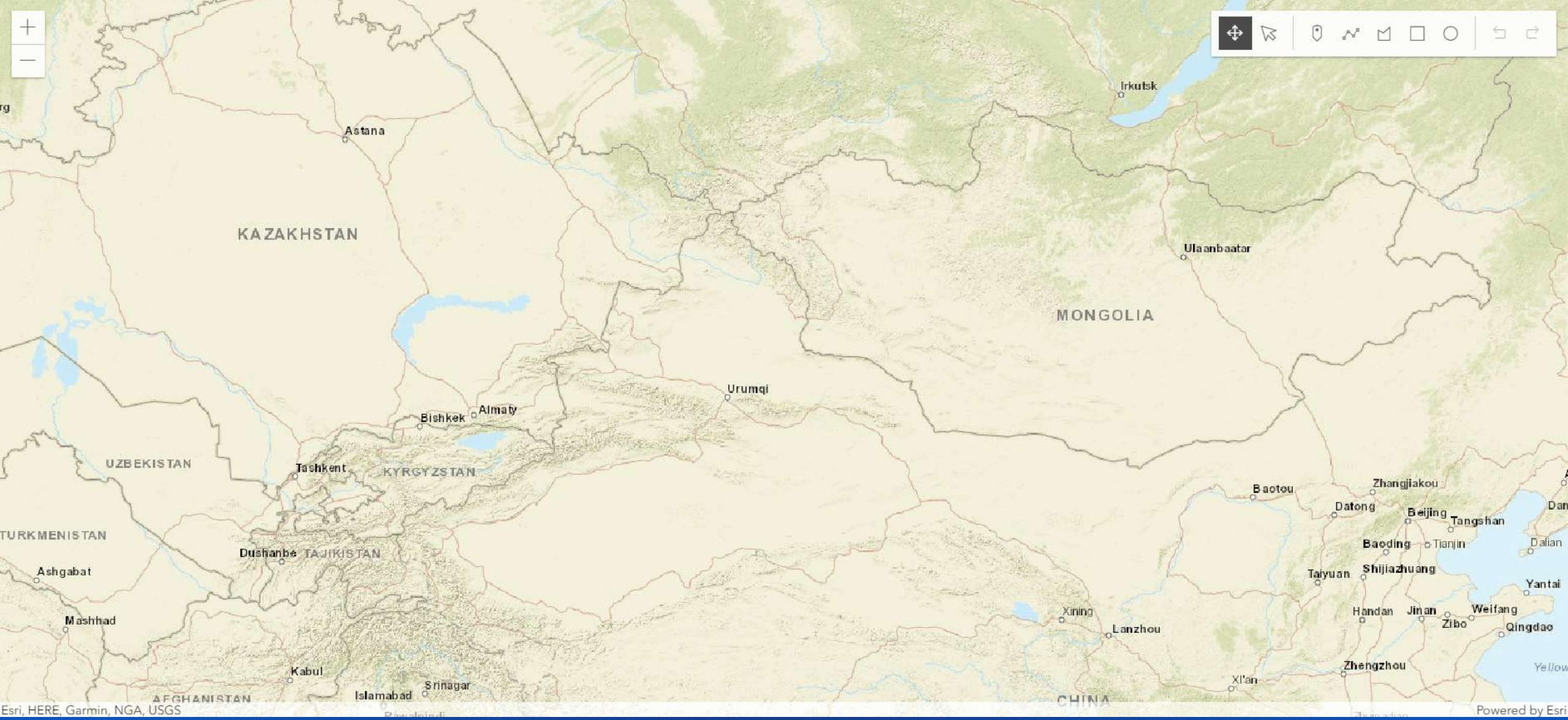


Widget development

Widgets are reusable user-interface components and are key to providing a rich user experience. The ArcGIS for JavaScript API provides a set of ready-to-use widgets. Beginning with version 4.2, it also provides a foundation for you to create custom widgets.

ViewModel pattern

There are two parts to working with the widget framework. These are: 1) the widget, and 2) the widget's ViewModel. The [Widget](#) (i.e. View), part is responsible for handling the User Interface (UI) of the widget, meaning how the widget displays and handles user interaction via the DOM. The ViewModel part is responsible for the underlying functionality of the widget, or rather, its business logic.



SKETCHING

- Draw graphics on the graphics layer
- Use the OOB widget

Issue status

In Progress

E.g. submitted, received, in progress, or completed.

Point of contact information

Who should we contact regarding this problem?

First name

Trystan

Last name

Mccoy

Telephone number

761-616-9091

Email

Update assessment

W St

Pacific Hwy

Em St

Soap Shop

3rd Ave

Tal

St

16th St

G St

1A

F St

Orange Line

Martin Luther King Jr Fwy

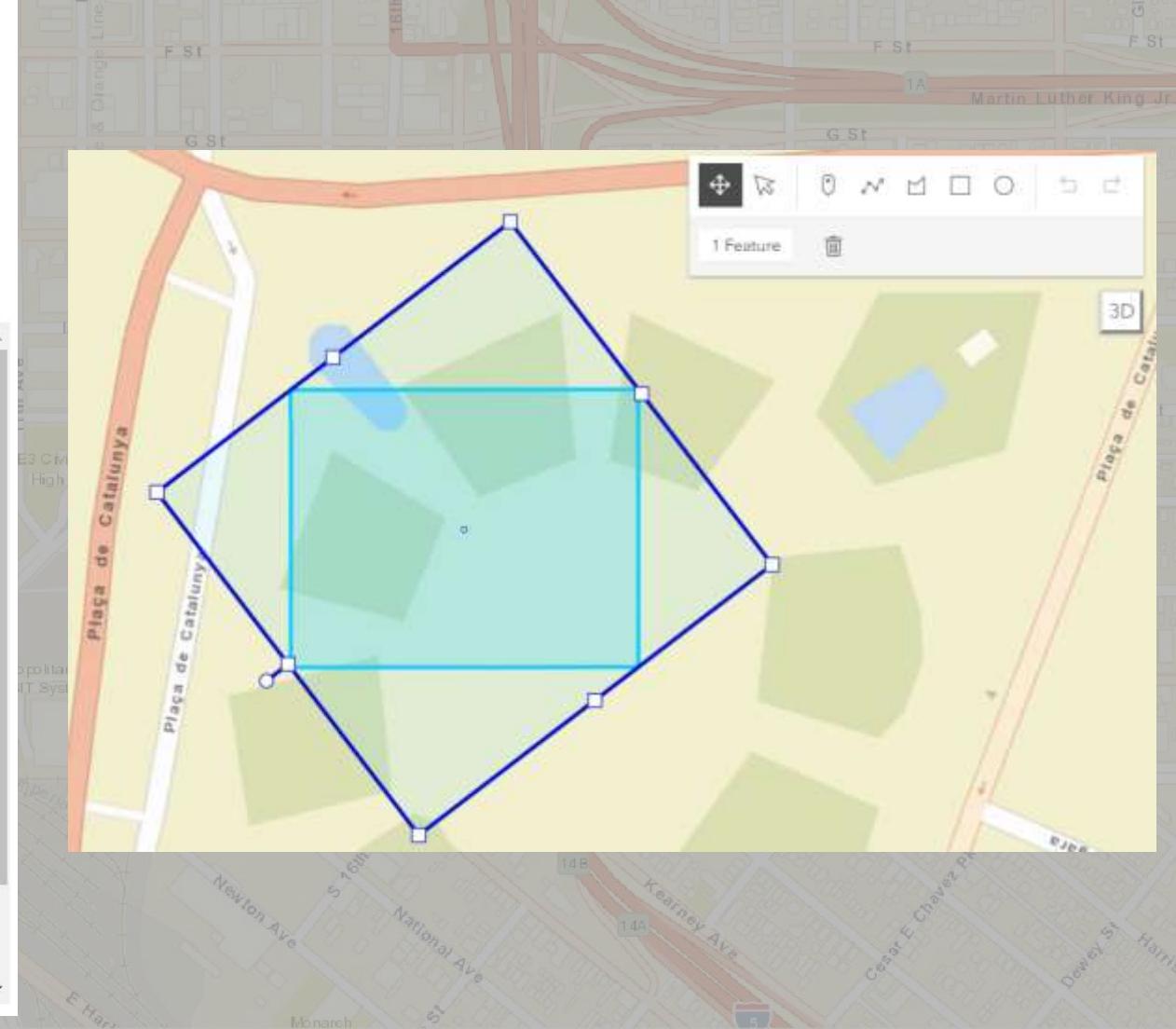
Report Incidents

- Select template from the list
- Click on the map to create a new feature
- Update associated attribute data
- Click *Update Incident Info*

Filter types

IncidentsReport - Incidents report

- Dead animal
- Graffiti
- Manhole cover
- Other
- Pothole
- Street light



EDITING

Form-based editing
Feature templates
Create & update geometry

Editor | API Reference

Search API Reference

- > esri
- > esri/core
- > esri/core/AccessorSupport
- > esri/core/workers
- > esri/geometry
- > esri/geometry/support
- > esri/identity
- > esri/layers
- > esri/layers/buildingSublayers

Editor

[Constructors](#) | [Properties](#) | [Methods](#) | [Type definitions](#)

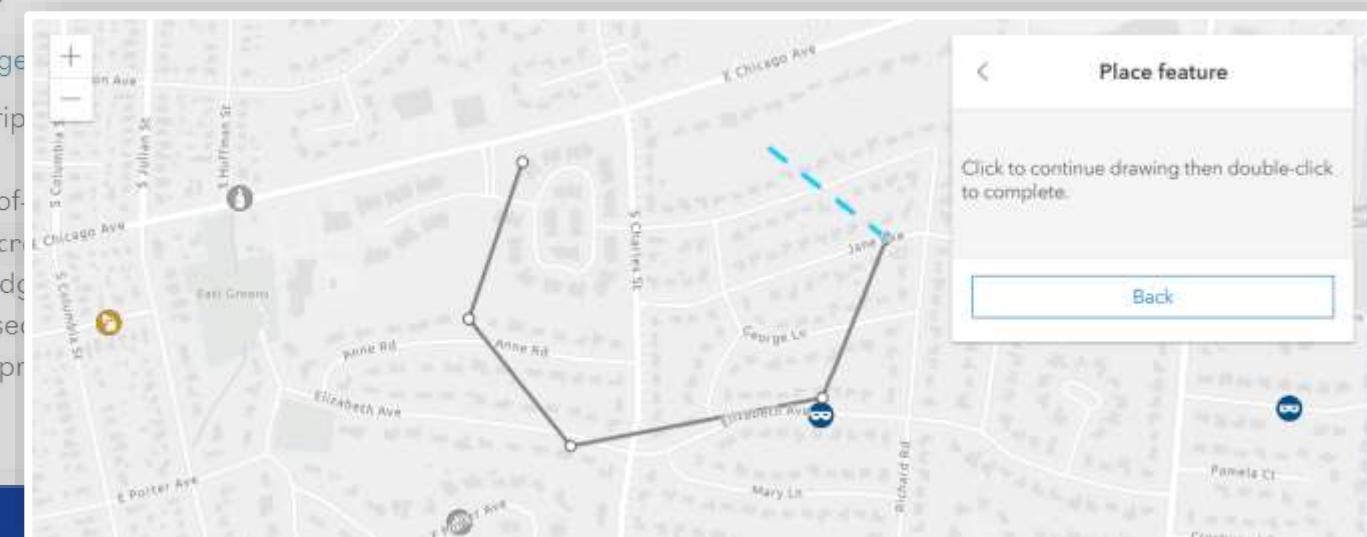
```
require(["esri/widgets/Editor"], function(Editor) { /* code goes here */ });
```

Class: [esri/widgets/Editor](#)

Inheritance: Editor → Widget

Since: ArcGIS API for JavaScript 4.1

This widget provides an out-of-the-box solution for editing features in an application. It has two different Workflow, creating and editing, modes. It can be used with one or more editable feature layer. The widget can be used with multiple layers. If the layers are not all editable, the layers can be used to define which layers are available for editing. The editingConfig property. This property defines the configuration for each layer that is used by the widget.



EDITOR WIDGET

Widget that brings together the editing experience

Use client-side geometric operations

Buffer, cut, merge, validation workflows, etc...

Method Overview

Name	Return Type	Summary	more details	Object
<code>buffer()</code>	<code>Polygon Polygon[]</code>	Creates planar (or Euclidean) buffer polygons at a specified distance around the input geometries.	more details	<code>geometryEngine</code>
<code>clip()</code>	<code>Geometry</code>	Calculates the clipped geometry from a target geometry by an envelope.	more details	<code>geometryEngine</code>
<code>contains()</code>	<code>Boolean</code>	Indicates if one geometry contains another geometry.	more details	<code>geometryEngine</code>
<code>convexHull()</code>	<code>Geometry Geometry[]</code>	Calculates the convex hull of the input geometry.	more details	<code>geometryEngine</code>
<code>crosses()</code>	<code>Boolean</code>	Indicates if one geometry crosses another geometry.	more details	<code>geometryEngine</code>
<code>cut()</code>	<code>Geometry[]</code>	Split the input Polyline or Polygon where it crosses a cutting Polyline.	more details	<code>geometryEngine</code>
<code>densify()</code>	<code>Geometry</code>	Densify geometries by plotting points between existing vertices.	more details	<code>geometryEngine</code>
<code>difference()</code>	<code>Geometry Geometry[]</code>	Creates the difference of two geometries.	more details	<code>geometryEngine</code>
<code>disjoint()</code>	<code>Boolean</code>	Indicates if one geometry is disjoint (doesn't intersect in any way) with another geometry.	more details	<code>geometryEngine</code>



ArcGIS API for JavaScript

Everything you need to build a compelling location experience for your business

[Get Started](#)

Tutorials

Use tutorials to start building an app with the ArcGIS API for JavaScript.

Guide

Learn how to do mapping, geocoding, routing, and other spatial analytics.

Sample Code

Get code samples for mapping, visualization, and spatial analysis.

API Reference

Documentation for all ArcGIS API for JavaScript classes, methods, and properties.

Showcase

See how to combine functionality into interactive and compelling applications.

Version 4.12 - July 2019 - Looking for v3.29?

[Get the API](#)[What's new](#)[Licensing](#)

Tooling





Using Frameworks

Overview
Release notes
Get the API
Quick Start
> Tutorials
> Core Concepts
> Data Visualization
> Building your UI
> Working with ArcGIS Online and Enterprise
▼ Developer Tooling
Using Frameworks
Using npm
Using webpack
Using PhoneGap
TypeScript Setup

The ArcGIS API for JavaScript has all the tools you would need to build fully scalable and effective applications. However, you may want to utilize another framework's specific capabilities, or leverage your in house expertise in a particular framework.

Frameworks and Libraries

There are many examples of integrating the ArcGIS API for JavaScript with popular frameworks such as [React](#), [Angular](#), [Vue](#), [Ember](#), and many others. Some frameworks and libraries are easier to integrate with than others, so below we introduce some tools and methods to assist you.

You can approach this framework integration in one of two ways.

1. Integrate a framework into your ArcGIS API for JavaScript application.
2. Integrate the ArcGIS API for JavaScript into an application built with a framework.

Map Centric Integration

In the first scenario, you may want to leverage a framework to help you build UI components to use with your application, but the map is still the main focus of your application and your development efforts. The ArcGIS API for JavaScript framework samples demonstrate how to take advantage of features such as view models to make it easy to use components from your framework of choice in an application that is built following the conventions of the ArcGIS API for JavaScript.

Examples:

Content
Frameworks and Libraries
Map Centric Integration
Framework First Integration
Modern JavaScript Development
Module Loading
@arcgis/webpack-plugin
esri-loader
Ember's loader.js

Plan your week...

<http://esriurl.com/uc2019webdev>



Products

Industries

Support & Services

About

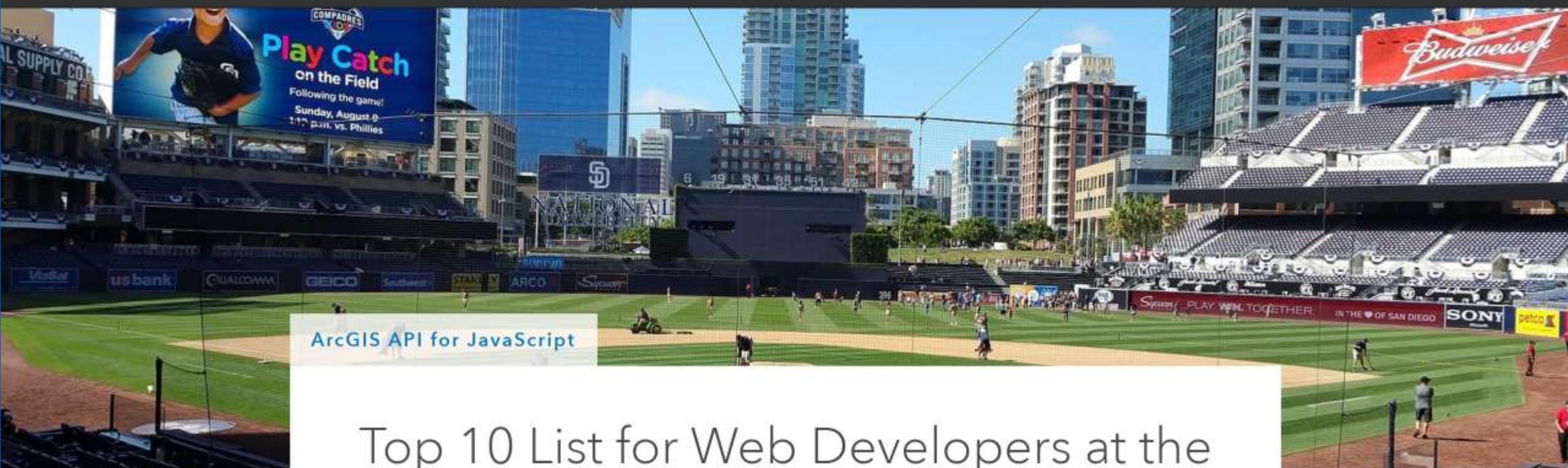


Sign In

[ArcGIS Blog](#)

Overview

Topics ▾



Top 10 List for Web Developers at the 2019 Esri User Conference

Announcements

June 24, 2019



Julie Powell,
Amy Niessen



Share your apps and suggestions with us...

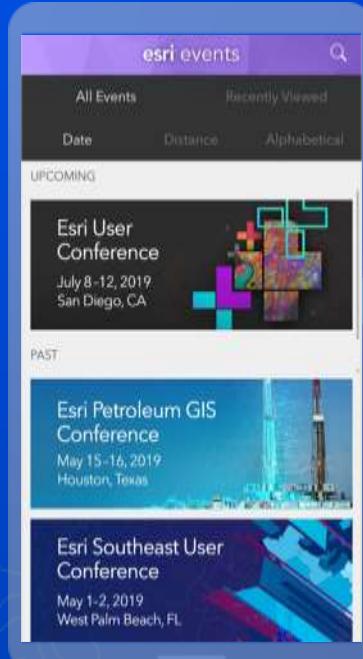
- Your apps!
- Your impressions on the latest API
- Ideas for next UC or Developer Summit related to web development



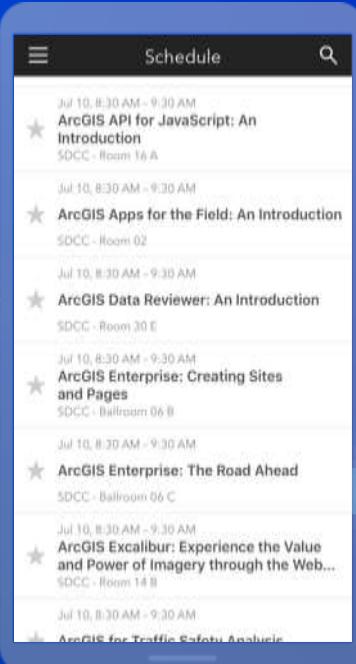
jsapi_pm@esri.com

Please Share Your Feedback in the App

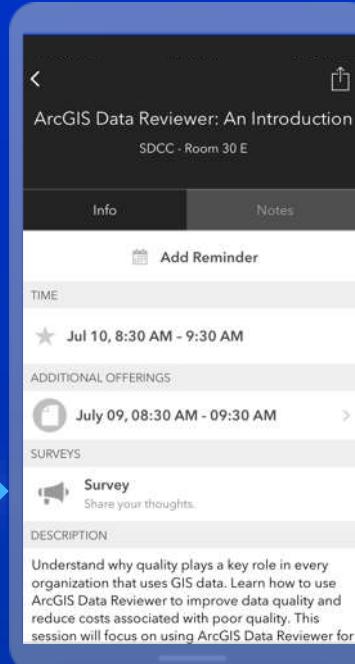
Download the Esri Events app and find your event



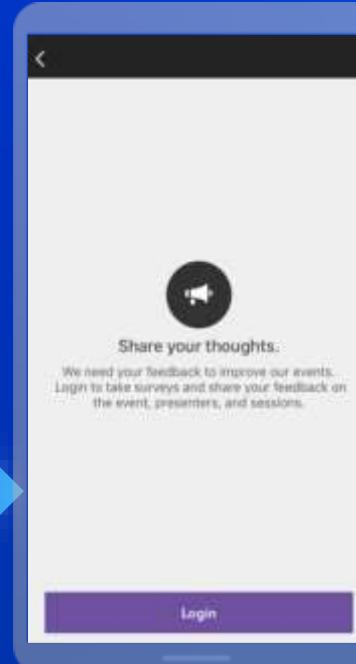
Select the session you attended



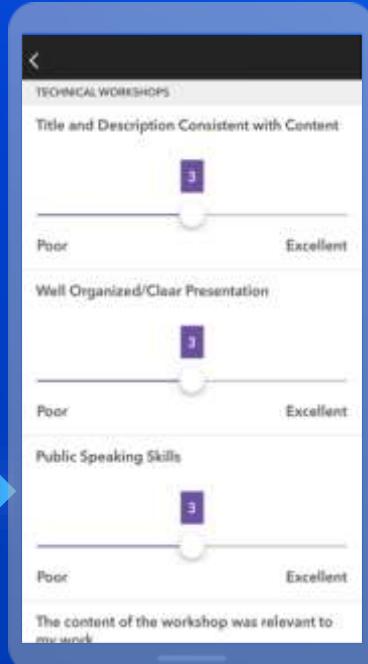
Scroll down to "Survey"



Log in to access the survey



Complete the survey and select "Submit"



Section Header

Section Subhead



Demo Title

Presenter(s)