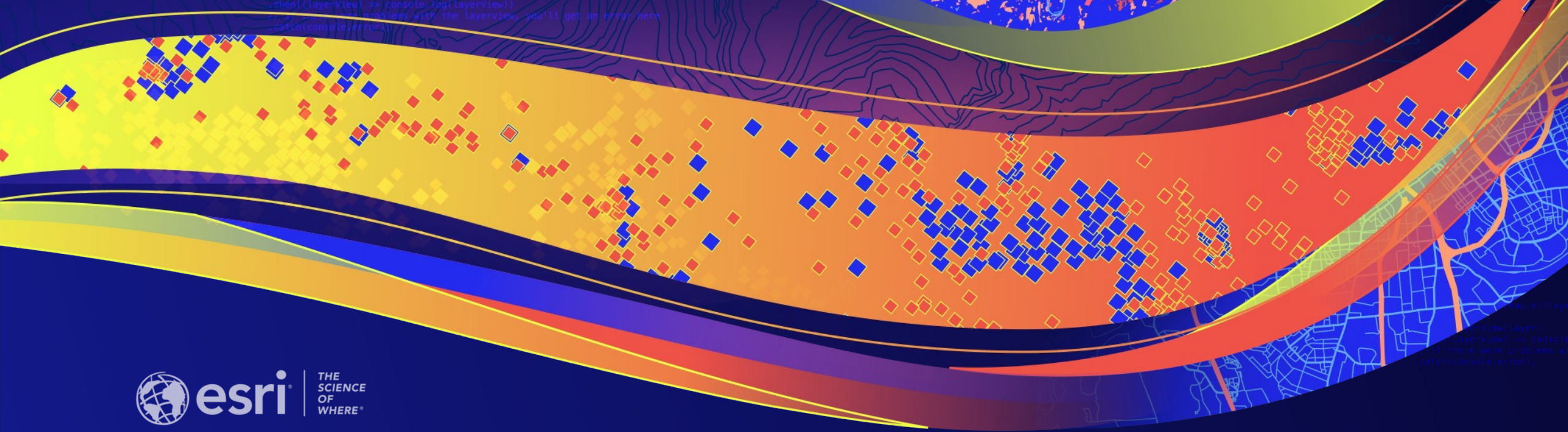


esri | THE SCIENCE OF WHERE

ESRI DEVELOPER SUMMIT 2023

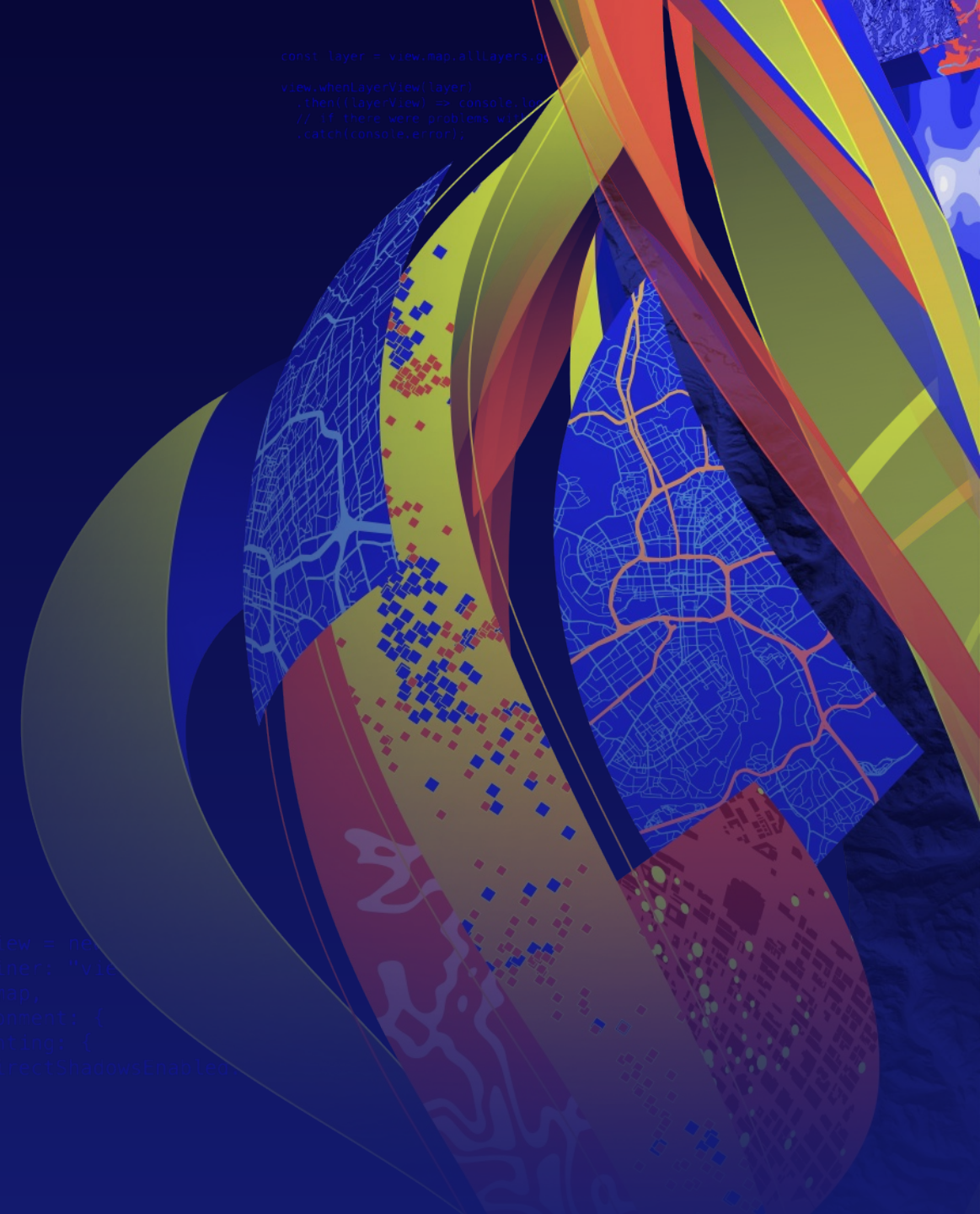
ArcGIS Maps SDK for JavaScript: Programming Patterns and API Fundamentals

Lauren Boyd and Sage Wall

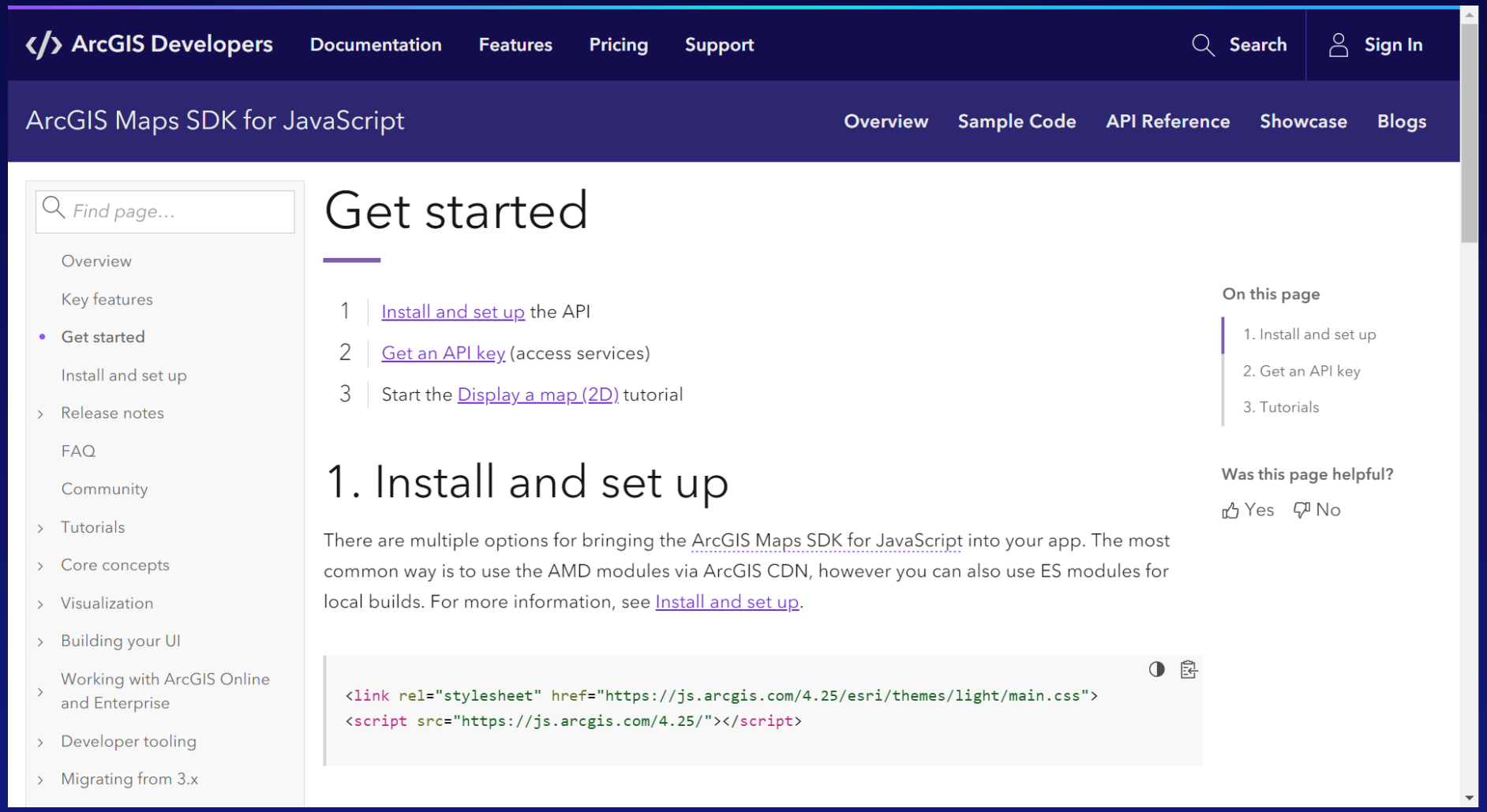
Presentation Resources: https://links.esri.com/DS23-programming-patterns

Accessing the SDK

Sage Wall



Get Started

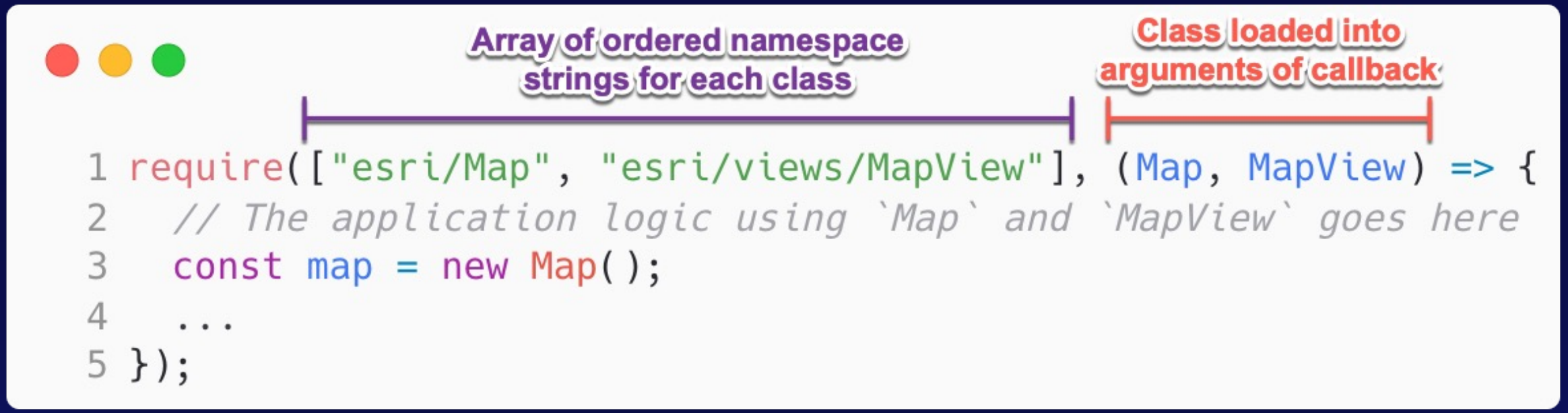

ArcGIS Developers
Documentation
Features
Pricing
Support
Search
Sign In
ArcGIS Maps SDK for JavaScript
Overview
Sample Code
API Reference
Showcase
Blogs
Find page...
Overview
Key features
Get started
Install and set up
Release notes
FAQ
Community
Tutorials
Core concepts
Visualization
Building your UI
Working with ArcGIS Online and Enterprise
Developer tooling
Migrating from 3.x
Get started
1 Install and set up the API
2 Get an API key (access services)
3 Start the Display a map (2D) tutorial
On this page
1. Install and set up
2. Get an API key
3. Tutorials
Was this page helpful?
Yes
No
1. Install and set up
There are multiple options for bringing the ArcGIS Maps SDK for JavaScript into your app. The most common way is to use the AMD modules via ArcGIS CDN, however you can also use ES modules for local builds. For more information, see Install and set up.
<link rel="stylesheet" href="https://js.arcgis.com/4.25/esri/themes/light/main.css">
<script src="https://js.arcgis.com/4.25/"></script>

Asynchronous Module Definition (AMD)

AMD Modules via ArcGIS CDN

```
<link rel="stylesheet" href="https://js.arcgis.com/4.26/esri/themes/light/main.css">
<script src="https://js.arcgis.com/4.26/"></script>
```

Loading Classes via AMD



ECMAScript Modules (ESM)

ES modules via NPM

```
npm install @arcgis/core
```

```
import Map from "@arcgis/core/Map.js";
import MapView from "@arcgis/core/views/MapView.js";

const map = new Map({
  basemap: "topo-vector"
});
```

Constructors, properties and methods

Classes have a constructor; property values can be set by passing parameters to the constructor

```
const map = new Map({
  basemap: "topo-vector"
});
```

Then the class's properties and methods are available

```
// sets the basemap property to "streets-vector"
map.basemap = "streets-vector";
```

```
// destroys the map and any associated resources
map.destroy();
```

Autocasting

Without Autocasting

```
import Color from "@arcgis/core/Color.js";
import Graphic from "@arcgis/core/Graphic.js";
import SimpleLineSymbol from "@arcgis/core/symbols/SimpleLineSymbol.js";
import SimpleFillSymbol from "@arcgis/core/symbols/SimpleFillSymbol.js";

const graphic = new Graphic({
  symbol: new SimpleFillSymbol({
    color: new Color([173, 216, 230, 0.2]),
    outline: new SimpleLineSymbol({
      color: new Color([255, 255, 255]),
      width: 1
    })
  })
});
```

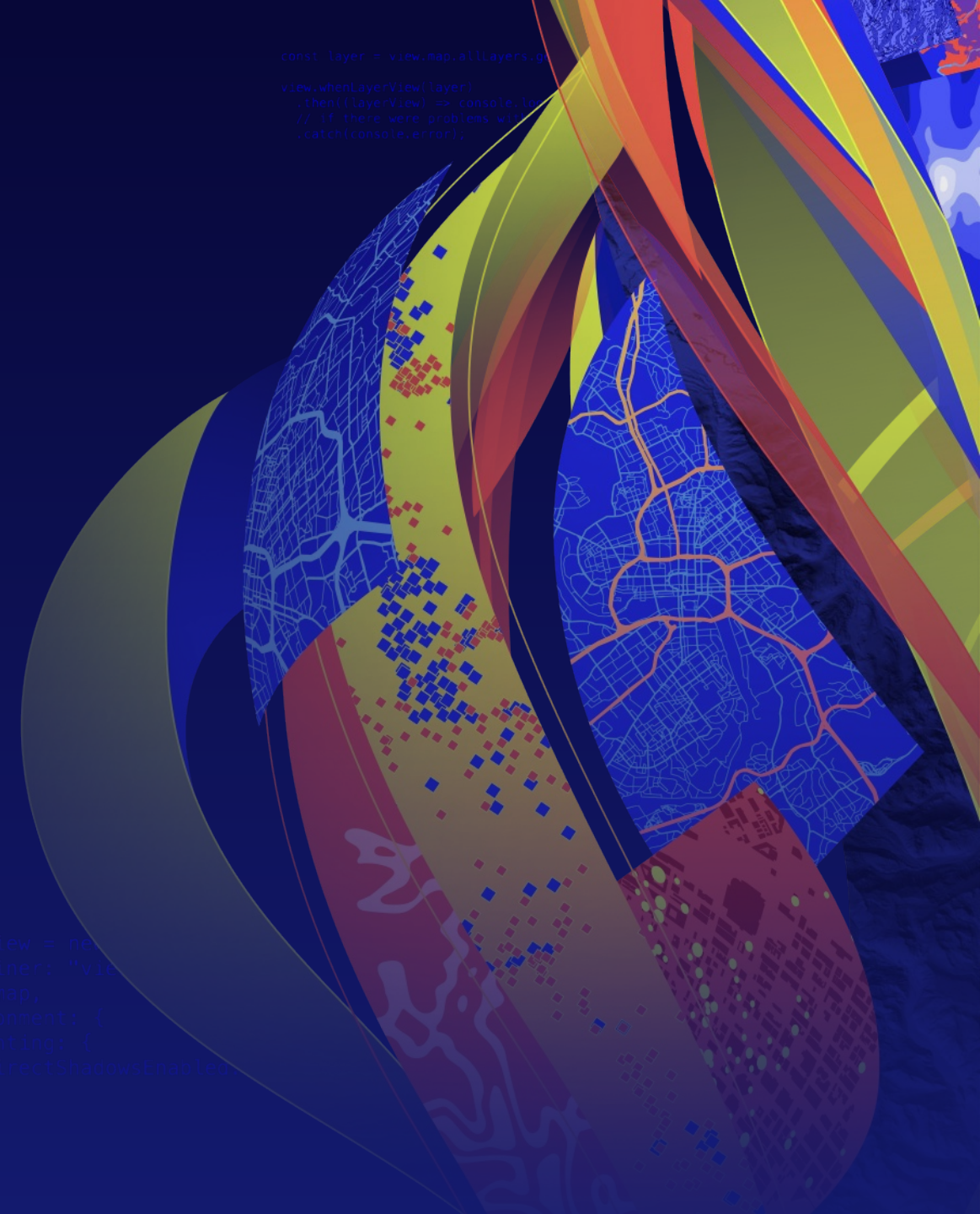
symbol Symbol autocast

With Autocasting

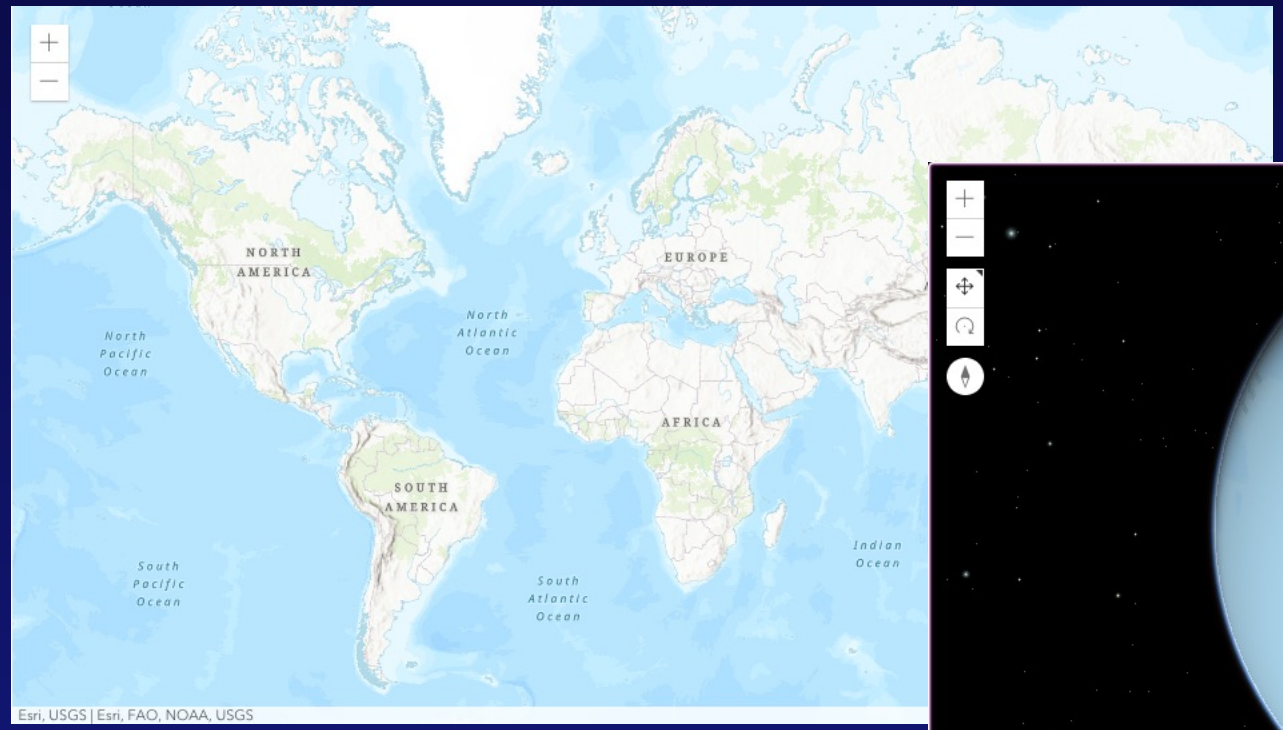
```
import Graphic from "@arcgis/core/Graphic.js";

const graphic = new Graphic({
  symbol: {
    type: "simple-fill", // autocasts as SimpleFillSymbol
    color: [173, 216, 230, 0.2],
    outline: {
      color: [255, 255, 255],
      width: 1
    }
  }
});
```

Maps and Views

Lauren Boyd



MapViews & SceneViews



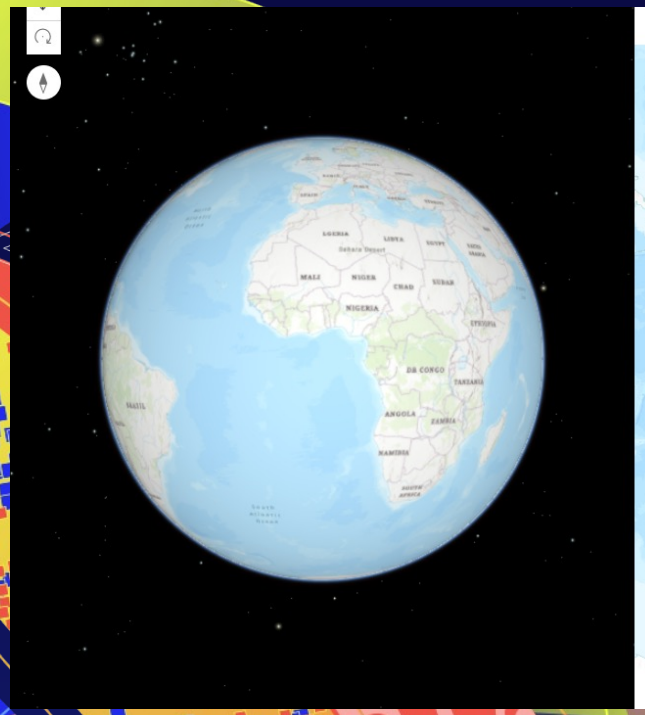
Map and Views

```
const map = new Map({
  basemap: "topo-vector",
  // World elevation service
  ground: "world-elevation"
});

const mapView = new MapView({
  map: map,
  container: "viewDiv"
});

const sceneView = new SceneView({
  map: map,
  container: "viewDiv"
});
```

Basemaps

- Basemaps for developers
  - For use with API keys
- Basemaps for ArcGIS Organization users
  - For use with ArcGIS Online/ArcGIS Enterprise users

```
const map = new Map({
  /*
  For ArcGIS Organizational Accounts:
    "satellite", "hybrid", "terrain", "oceans", "osm",
    "dark-gray-vector", "gray-vector", "streets-vector",
    "topo-vector", "streets-night-vector", "streets-relief-vector",
    "streets-navigation-vector"
  For use with API keys:
    "arcgis-imagery", "arcgis-imagery-standard", "arcgis-imagery-labels",
    "arcgis-light-gray", "arcgis-dark-gray", "arcgis-navigation",
    "arcgis-navigation-night", "arcgis-streets", "arcgis-streets-night",
    "arcgis-streets-relief", "arcgis-topographic", "arcgis-oceans",
    "osm-standard", "osm-standard-relief", "osm-streets",
    "osm-streets-relief", "osm-light-gray", "osm-dark-gray",
    "arcgis-terrain", "arcgis-community", "arcgis-charted-territory",
    "arcgis-colored-pencil", "arcgis-nova", "arcgis-modern-antique",
    "arcgis-midcentury", "arcgis-newspaper", "arcgis-hillshade-light",
    "arcgis-hillshade-dark", "arcgis-human-geography", "
    arcgis-human-geography-dark"
  */
  basemap: "topo-vector",
  // World elevation service
  ground: "world-elevation"
})
```



Maps and Views

[ASM Demo]

[ESM Demo]

Layers and visualization

Lauren Boyd

Layers

Create a layer from a URL or portal item

```
const layer = new FeatureLayer({
  url: "https://services3.arcgis.com/GVgbJbqm8hXASVYi/arcgis/rest/services/Trailheads/FeatureServer/0",
  // portalItem: {
  //   id: "883cedb8c9fe4524b64d47666ed234a7",
  //   portal: "https://www.arcgis.com"              // Default: The ArcGIS Online Portal
  // }
});

map.add(layer);
```

Different layer types depending on your data source – see the Layer documentation for the full list

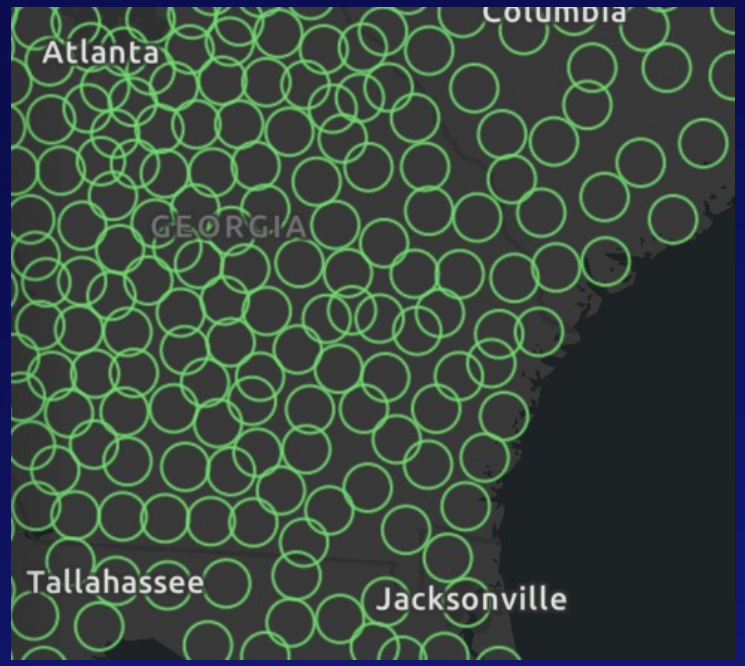
Visualization

Visualization Style	Renderer
Location Style	Simple Renderer
Data Driven Style	Unique Types, Class Breaks, Visual Variables, Time, Multivariate, Predominance, Dot Density, Relationship, Smart Mapping, Pie Charts
High Density Data	Clustering, Binning, Heatmap, Opacity, Bloom, Aggregation, Thinning, Visible Scale Range
3D Visualization	Global and local scenes, terrain renderering, cities in 3D, Visualizing points with 3D symbols

Renderers

- Renderers define how to visually represent features in the layer

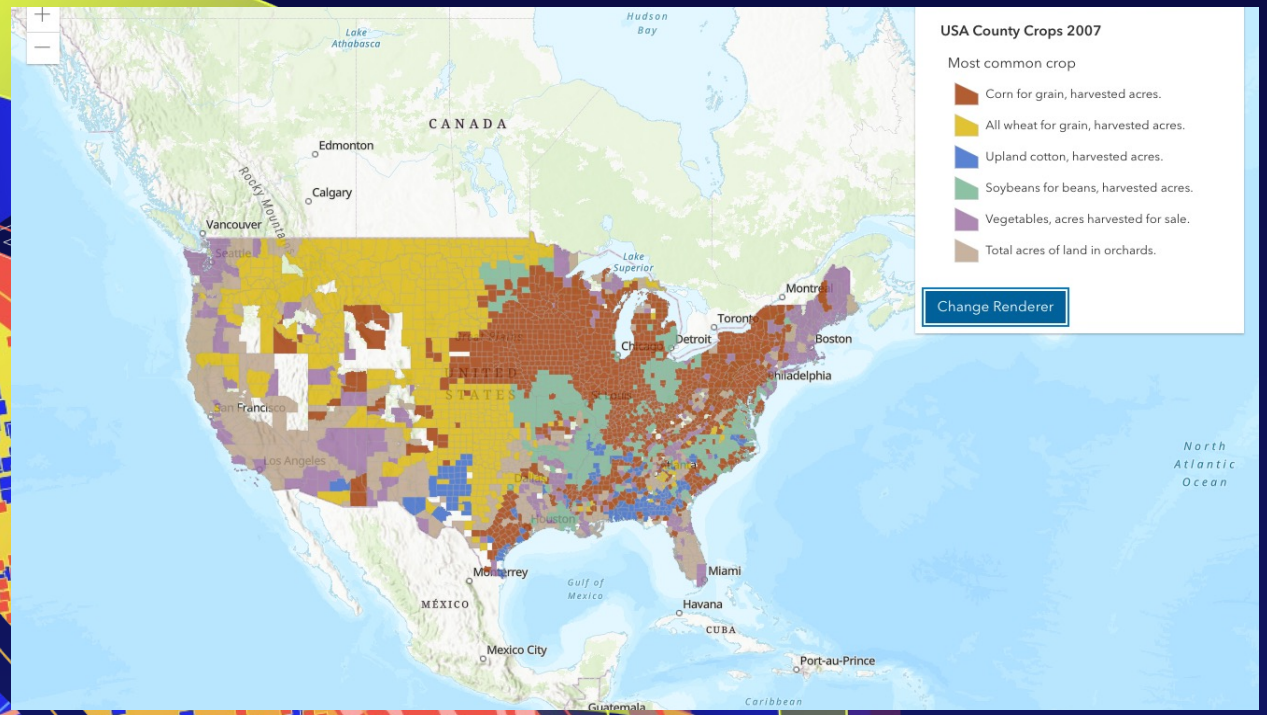
```
const layer = new FeatureLayer({
  url: "SERVICE URL",
  renderer: {
    type: "simple", // autocasts as new SimpleRenderer()
    symbol: {
      // define symbol properties
      type: "simple-marker", // autocasts as new SimpleMarkerSymbol()
      outline: {
        color: [0, 246, 84, 1]
      },
      size: 14,
      color: [255, 255, 255, 0]
    }
  }
})
```



Smart Mapping

- Easy way to generate visualizations
- Focus on:
  - One visualization method
  - Subset of attributes
  - Specific Area

```
let layer = new FeatureLayer({
  url:
"https://services.arcgis.com/V6ZHFr6zdgNZuVG0/arcgis/rest/services
/counties_politics_poverty/FeatureServer/0"
});

// simple visualization to indicate features with a single symbol
let params = {
  layer: layer,
  view: view
};

// when the promise resolves, apply the renderer to the layer
locationRendererCreator.createRenderer(params)
  .then(function(response){
    layer.renderer = response.renderer;
  });
```



Smartmapping

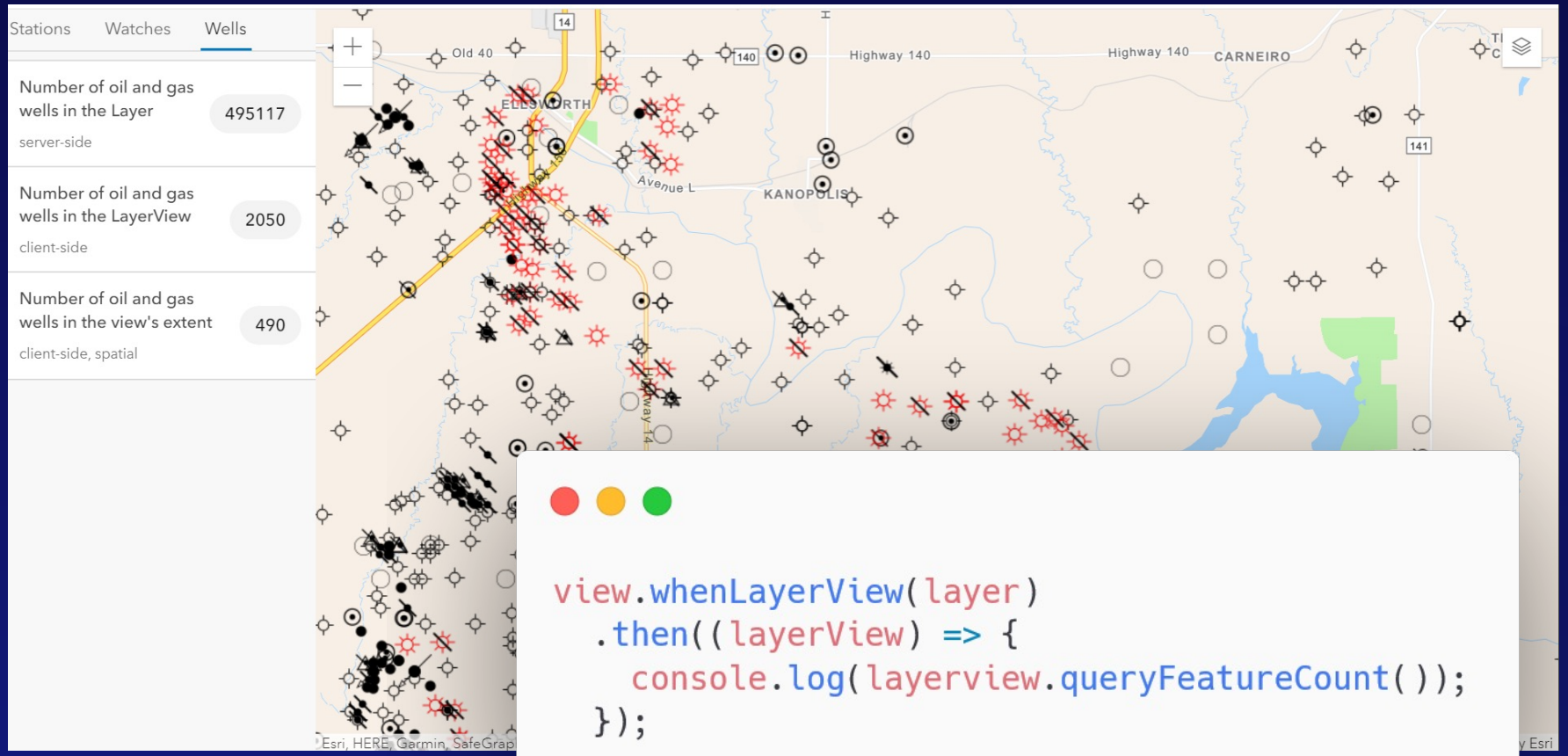
[Demo]

LayerViews and querying

Sage Wall

LayerView

- Renders features in the view
- Client-side queries, filtering, and effects




```
view.whenLayerView(layer)
  .then((layerView) => {
    console.log(layerview.queryFeatureCount());
  });
```

Creating and using a query

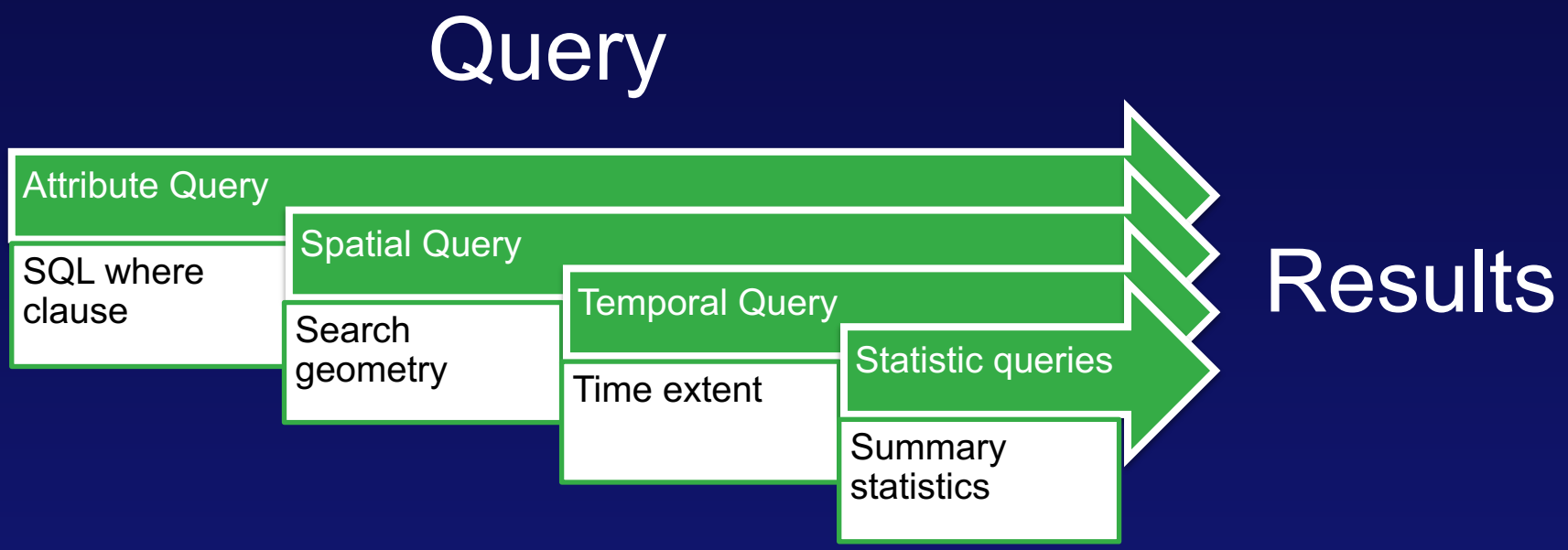
- Using **createQuery()**
  - Maintains filters and definition expressions
- **new Query()**
  - Clean query object

```
const query = featureLayer.createQuery();
query.where = "Name = 'Sage'";

featureLayer.queryFeatures(query)
  .then((results) => {
    console.log(results)
  });
```

Query Methods

queryFeatures()	• Queries features and returns a FeatureSet
queryExtent()	• Queries features and returns an extent
queryFeatureCount()	• Queries features and returns a count
queryObjectIds()	• Queries features and returns an array of objectIds

Query Types


Query
Attribute Query
SQL where clause
Spatial Query
Search geometry
Temporal Query
Time extent
Statistic queries
Summary statistics
Results

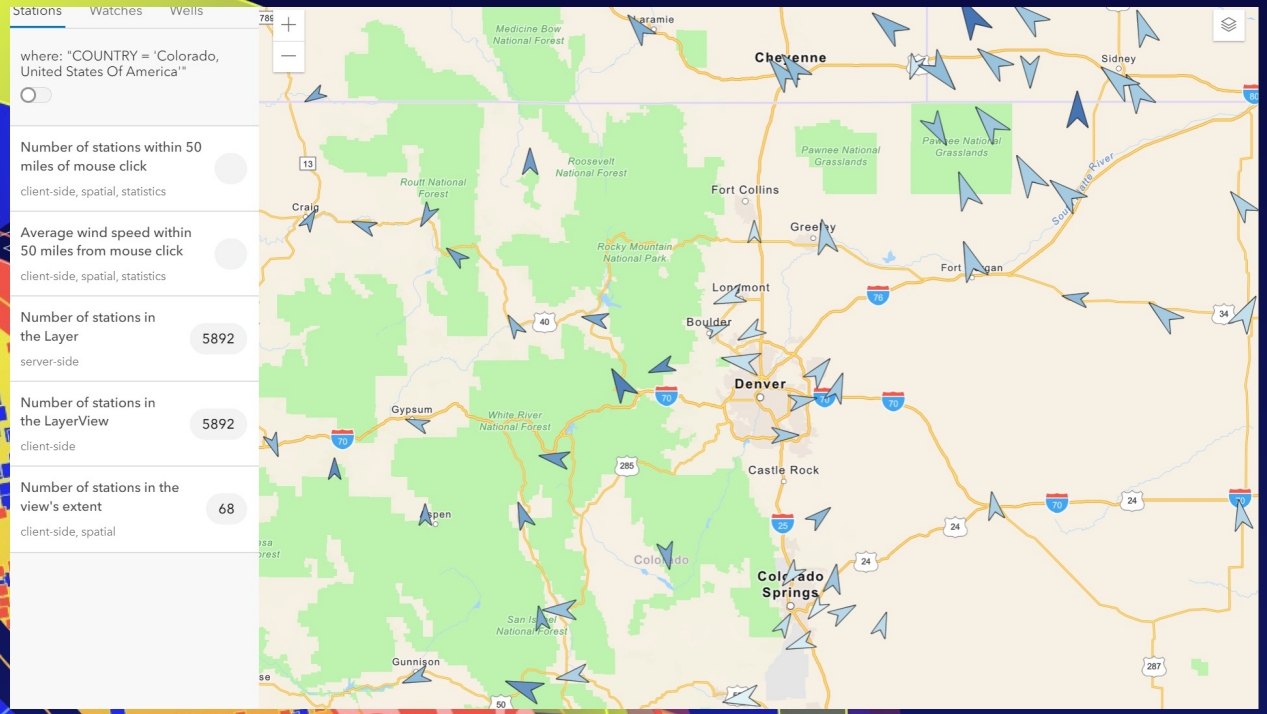
Should I use a layer query or a layer view query?

What's Important

Performance	• LayerView Query
Geographic precision	• Layer Query
Query every feature	• Layer Query

Tips when using client-side queries on layer views

```
// add fields to the layer's outFields properties
const stationsLayer = new FeatureLayer({
  outFields: ["OBJECTID", "COUNTRY", "TEMP", "WIND_DIRECT", "WIND_SPEED"],
  url
});

// wait for the layer view to be loaded in the view
const stationsLayerView = await view.whenLayerView(stationsLayer);

// wait for the layer view to stop updating
await reactiveUtils.whenOnce(
    () =>
      stationsLayerView.updating === false
  );

// query the number of features in the stations layer view
const stationsLayerViewCount = await stationsLayerView.queryFeatureCount();
```

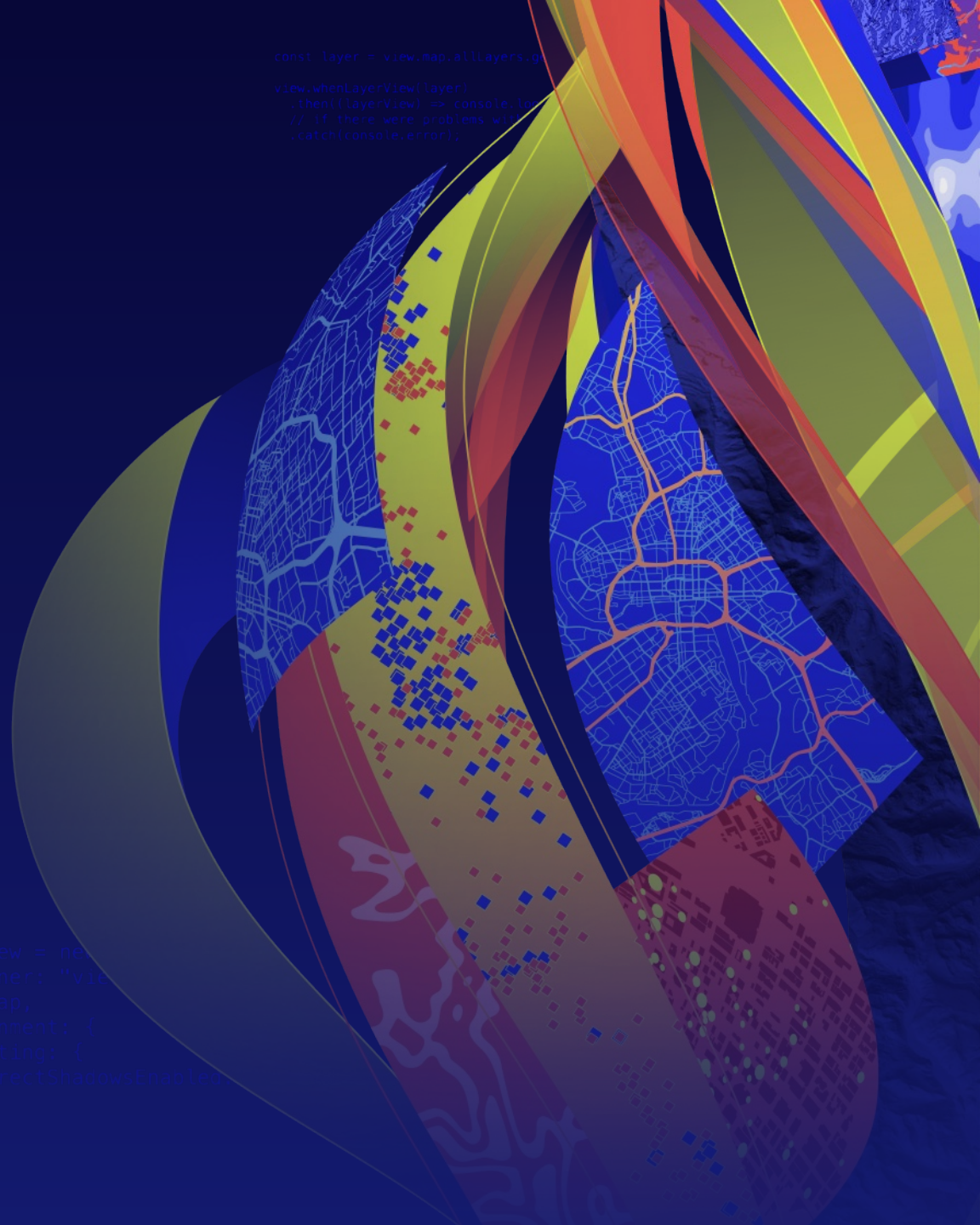


Querying and LayerView

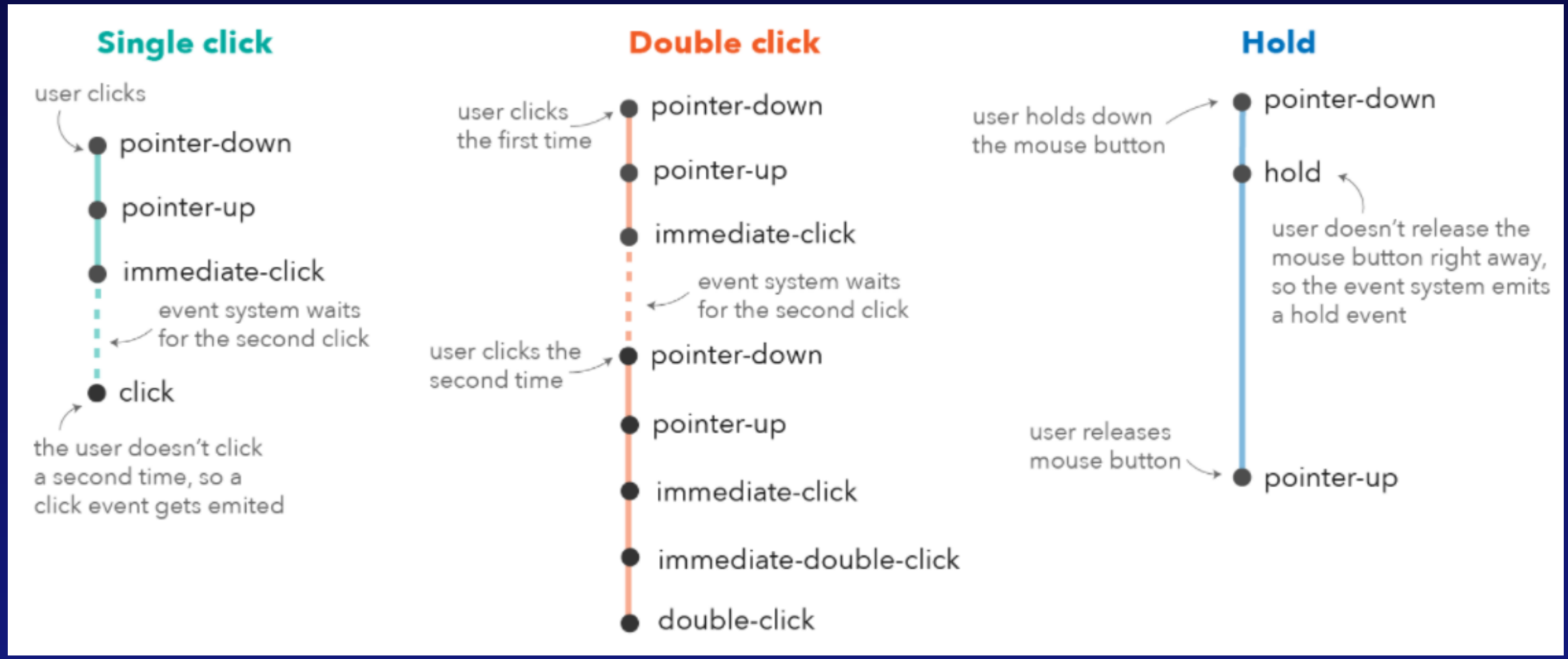
[Demo]

Event handling

Sage Wall



Events


Single click
user clicks
pointer-down
pointer-up
immediate-click
event system waits
for the second click
click
the user doesn't click
a second time, so a
click event gets emited
Double click
user clicks
the first time
pointer-down
pointer-up
immediate-click
event system waits
for the second click
user clicks the
second time
pointer-down
pointer-up
immediate-click
immediate-double-click
double-click
Hold
user holds down
the mouse button
pointer-down
hold
user doesn't release the
mouse button right away,
so the event system emits
a hold event
user releases
mouse button
pointer-up

Event Handling

```
// when view emits a click event
view.on("click", (event) => {
  view.graphics.removeAll();
  addGraphic(event);
});

// add a graphic to the view at the click event's location
const addGraphic = (screenPoint) => {
  const point = view.toMap(screenPoint);
  const newClickGraphic = clickGraphic.clone();
  newClickGraphic.geometry = point;
  view.graphics.add(newClickGraphic);
};
```

Event handling

[Demo]

Programming patterns

Lauren Boyd

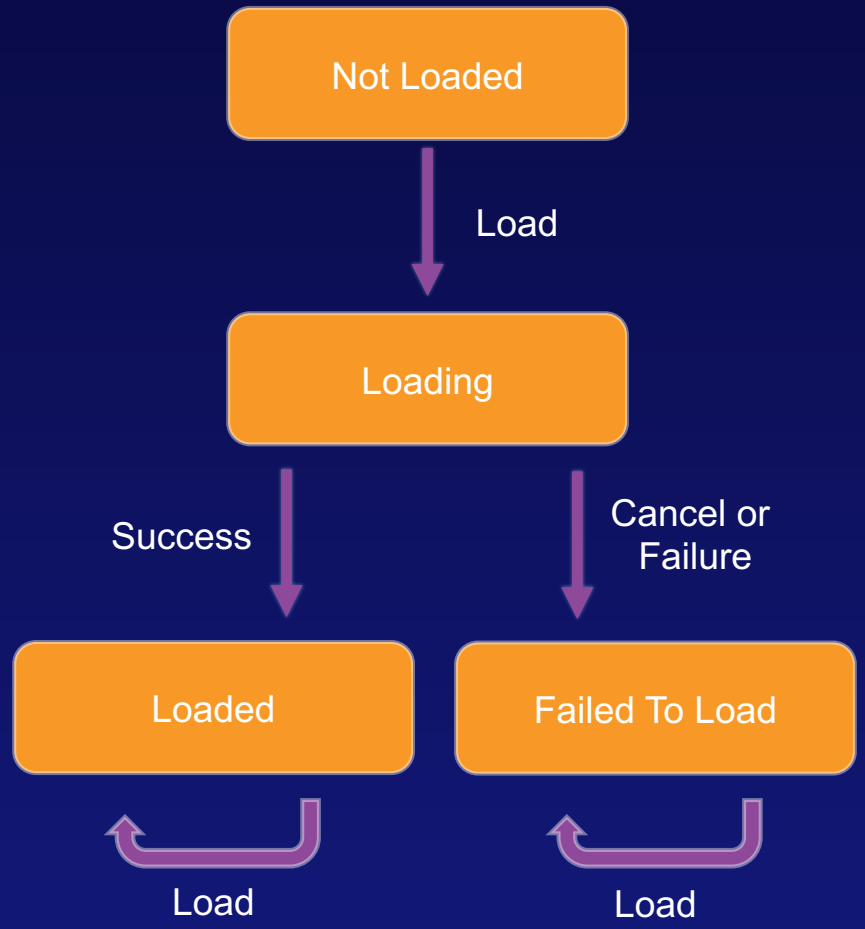
Promises

Promise = representation of a future value returned from an asynchronous task


```
1 // Create a feature layer from a url
2 let layer = new FeatureLayer(url);
3
4 map.add(layer);
5
6 view.whenLayerView(layer)
7     .then(function(layerView) {
8       // Do something with the layerview
9     })
10    .catch(function(error) {
11      // Error occurred creating the layerview
12    });
```

Loadable

```
const view = new MapView({
  container: "viewDiv"
});

const portal = new Portal({
  url: "https://myportal/"
});

const webmap = new WebMap({
  portalItem: {
    portal: portal,
    id: "f2e9b762544945f390ca4ac3671cfa72"
  }
});

webmap.load()
  .then(() => { view.map = webmap; })
  .catch((error) => {
    console.error("The resource failed to load: ", error);
  });
```



Reacting to Changes

- Watch when values change
- React when expressions become truthy

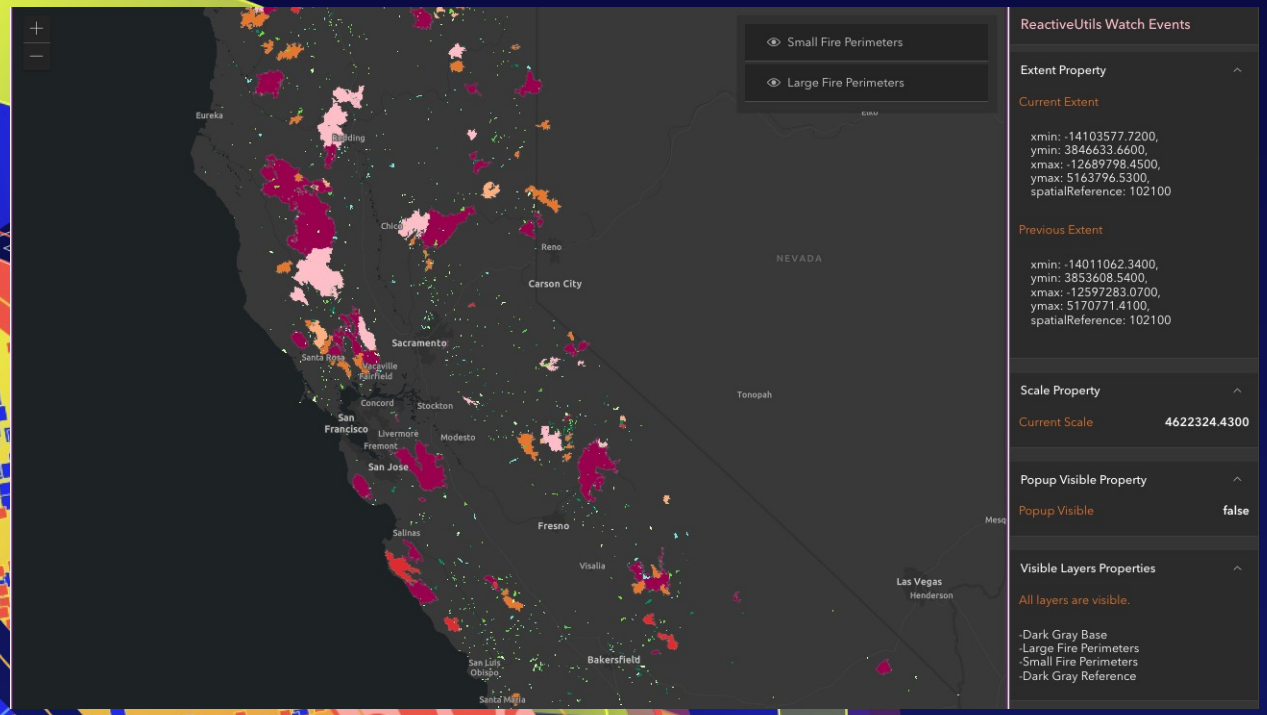
```
// React when an expression becomes truthy
reactiveUtils.when(
  () => view.scale > 1000,
  () => {
    console.log(`Scale is greater than 1000`);
  }
);

// Observe for when a boolean property becomes true
// Equivalent to watchUtils.whenTrue()
reactiveUtils.when(
  () => view?.stationary === true,
  async () => {
    console.log("User is no longer interacting with the map");
    await drawBuffer();
  }
);
```

Reacting to Changes

```
// React to multiple property changes
reactiveUtils.watch(
    () => [view.stationary, view.scale],
    ([stationary, scale]) => {
        console.log(`View is stationary: ${stationary} and scale is ${scale}`);
    });

// React to Collection changes
reactiveUtils.watch(
    () => view.map.layers((layer) => layer.title),
    (titles) => {
        console.log(`Layer titles changed! New titles: ${titles}`);
    });
```

- Combine values from multiple sources
- Watch for changes within a Collection

```
nst view = new SceneView({
container: "viewDiv",
map: map,
environment: {
  lighting: {
    directShadowsEnabled: true
  }
```

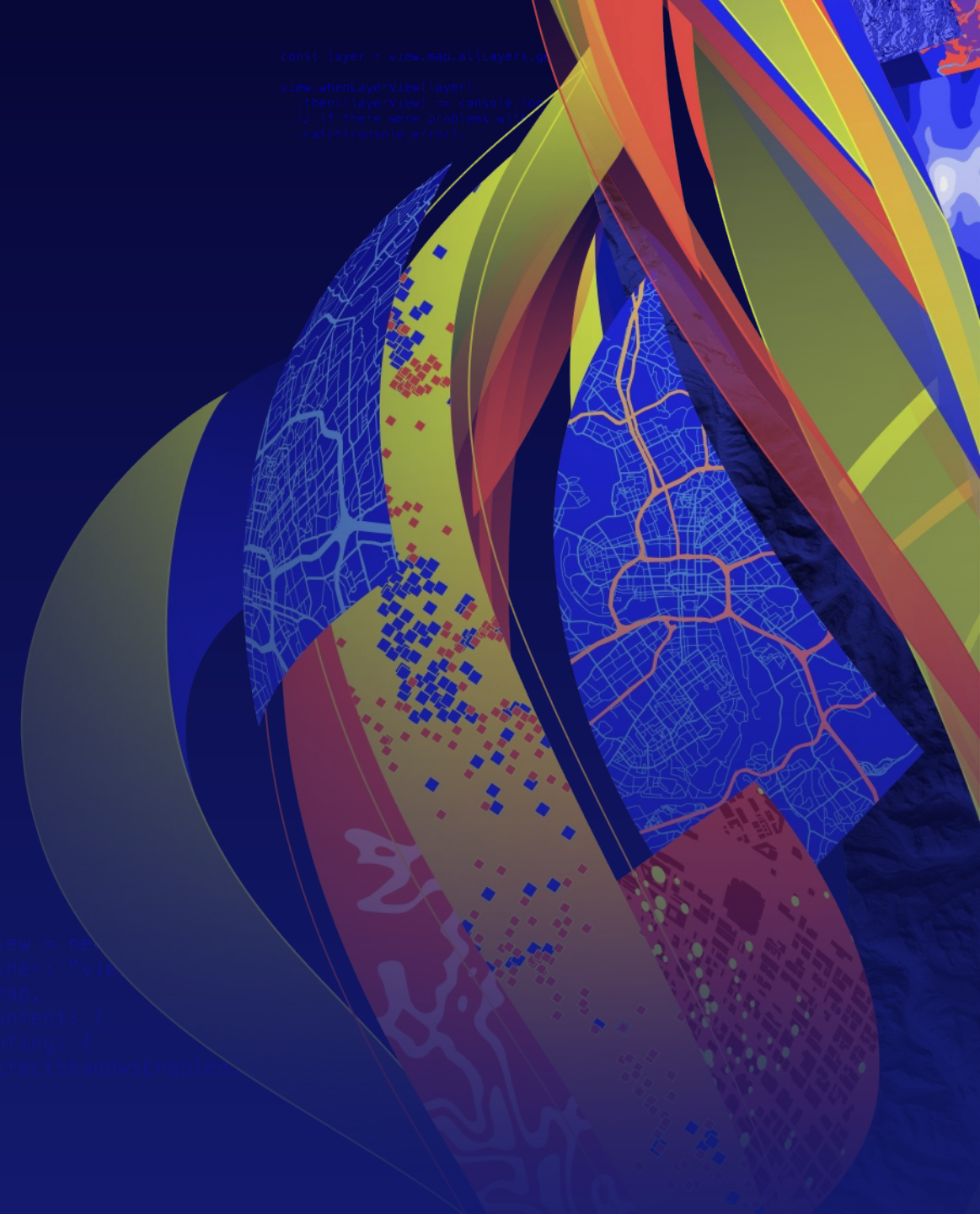


reactiveUtils

[Demo]

Widgets

Lauren Boyd

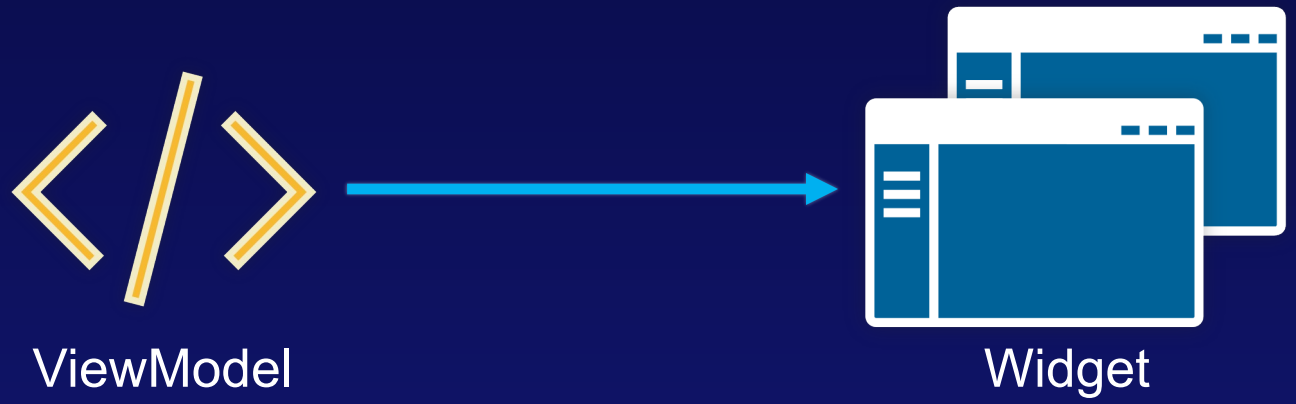


Widgets

- Professionally Designed and tested building blocks
- Responsive, Accessible, and Localized
- Out of the box

```
const widget = new SomeWidget({
  view,
  ...otherOptions
})
```

ViewModels



Weather

Demo


Editor
Settings
Edit features
Select
Create features
Filter types
Vegetation
Date Palm
California Palm
Tree
Airtransportation
Helicopter
Plane
Path
Street

Editor

[Demo]

Where can I learn more?

- SDK Documentation
- Programming Patterns Guide
- Esri Blogs
- Esri Community

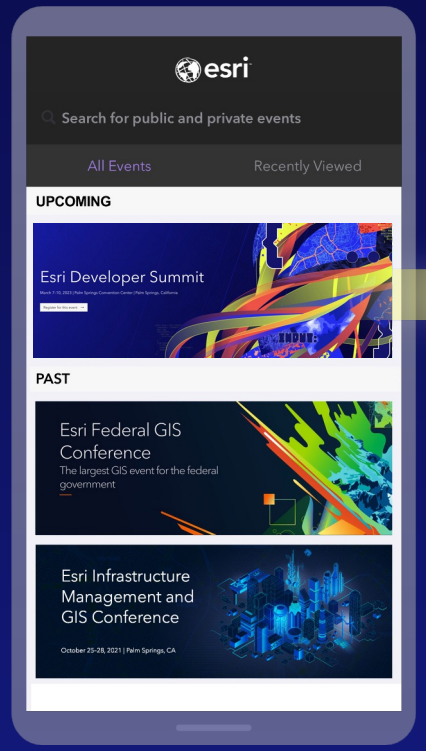
Presentation Resources: https://links.esri.com/DS23-programming-patterns

Upcoming Sessions

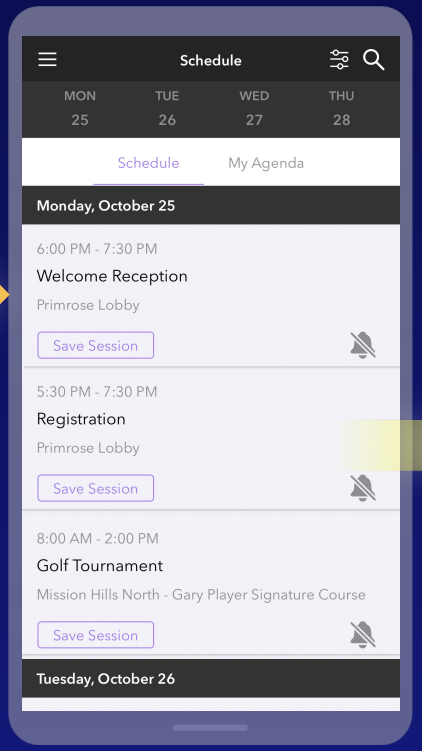
Session	Day/Time	Location
ArcGIS Maps SDK for JavaScript: Key Highlights of the Last Year	Tuesday, 5:30PM – 6:30PM	Primrose A | PSCC
ArcGIS Maps SDK for JavaScript: Data Visualization	Wednesday, 10:30PM – 11:30PM	Catalina/Madera | Renaissance Hotel
ArcGIS Online: Web Mapping with Arcade Expressions	Wednesday, 2:30PM – 3:30PM	Mojave Learning Center | Renaissance Hotel
ArcGIS Maps SDK for JavaScript: Using the FeatureLayer	Wednesday, 4:00PM-5:00PM	Catalina/Madera | Renaissance Hotel
ArcGIS Maps SDK for JavaScript: Bring in Data from Anywhere	Thursday, 10:30AM – 11:30AM	Mesquite C | PSCC
ArcGIS Maps SDK for JavaScript: Tips and Tricks for Developing and Debugging Apps	Thursday, 1:00PM – 1:30PM	Demo Theater 3: Mesquite D-E | PSCC
ArcGIS Maps SDK for JavaScript: Using Arcade with Your Apps	Thursday, 2:30PM – 3:30PM	Smoketree A-E | PSCC
Q&A with the ArcGIS Maps SDK for JavaScript Team	Friday, 8:30AM – 9:30AM	Pasadena/Sierra/Ventura
ArcGIS Maps SDK for JavaScript: the Road Ahead	Friday, 10:00AM – 11:00AM	Catalina/Madera

Please Share Your Feedback in the App

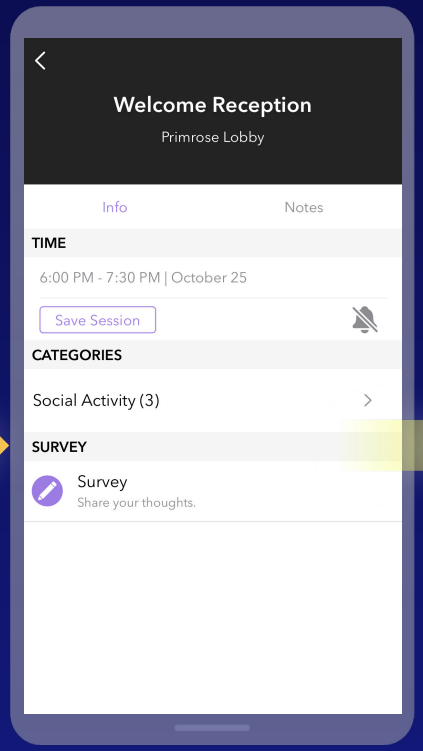
Download the Esri Events app and find your event



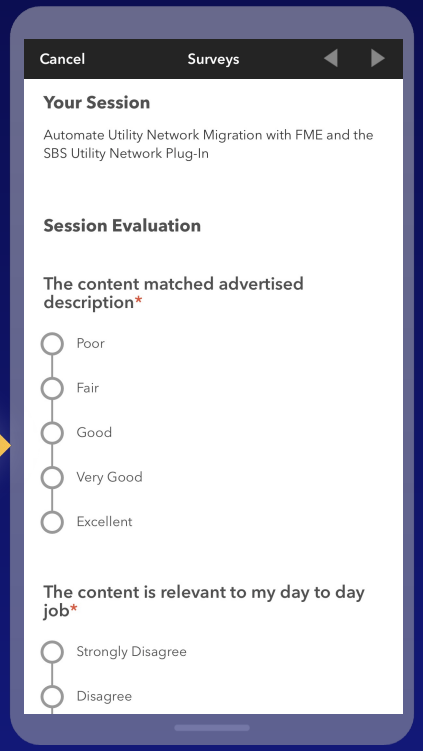
Select the session you attended



Scroll down to “Survey”



Log in to access the survey



Connect With Us On Social

And Join the Conversation Using #EsriDevSummit

- twitter.com/EsriDevs #esridevsummit, #esridevsummit23
- twitter.com/EsriDevEvents
- youtube.com/@EsriDevs
- links.esri.com/DevVideos
- github.com/Esri
- github.com/EsriDevEvents
- links.esri.com/EsriDevCommunity