esri | THE SCIENCE OF WHERE

2022 ESRI USER CONFERENCE

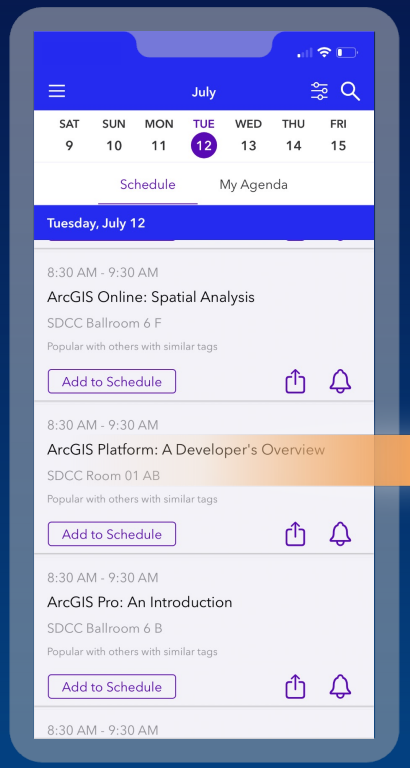
ArcGIS API for JavaScript: Programming Patterns and API Fundamentals

Anne Fitz, Lauren Boyd

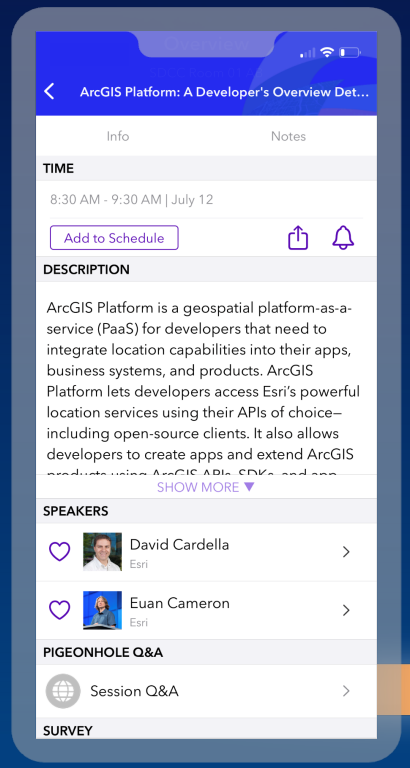
In-Person Digital Q&A Tool

for Technical Workshop, Demo Theater and User Presentation Sessions

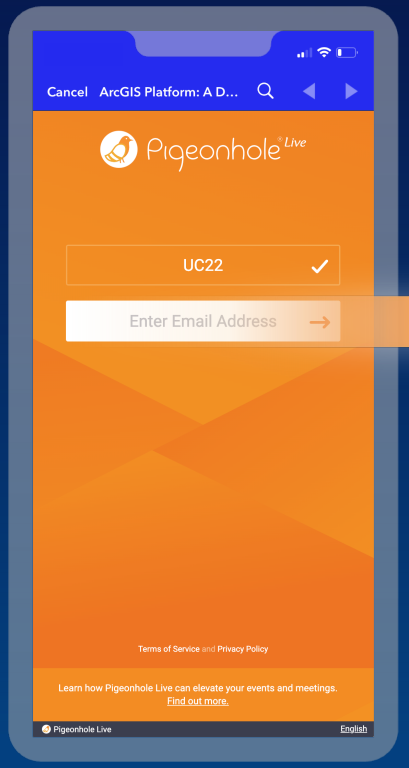
Select your session



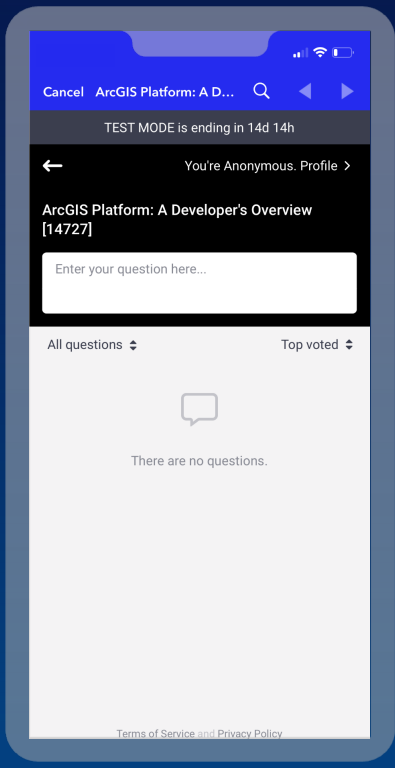
Click on the Pigeonhole Session Q&A link



Enter the email address used when registering



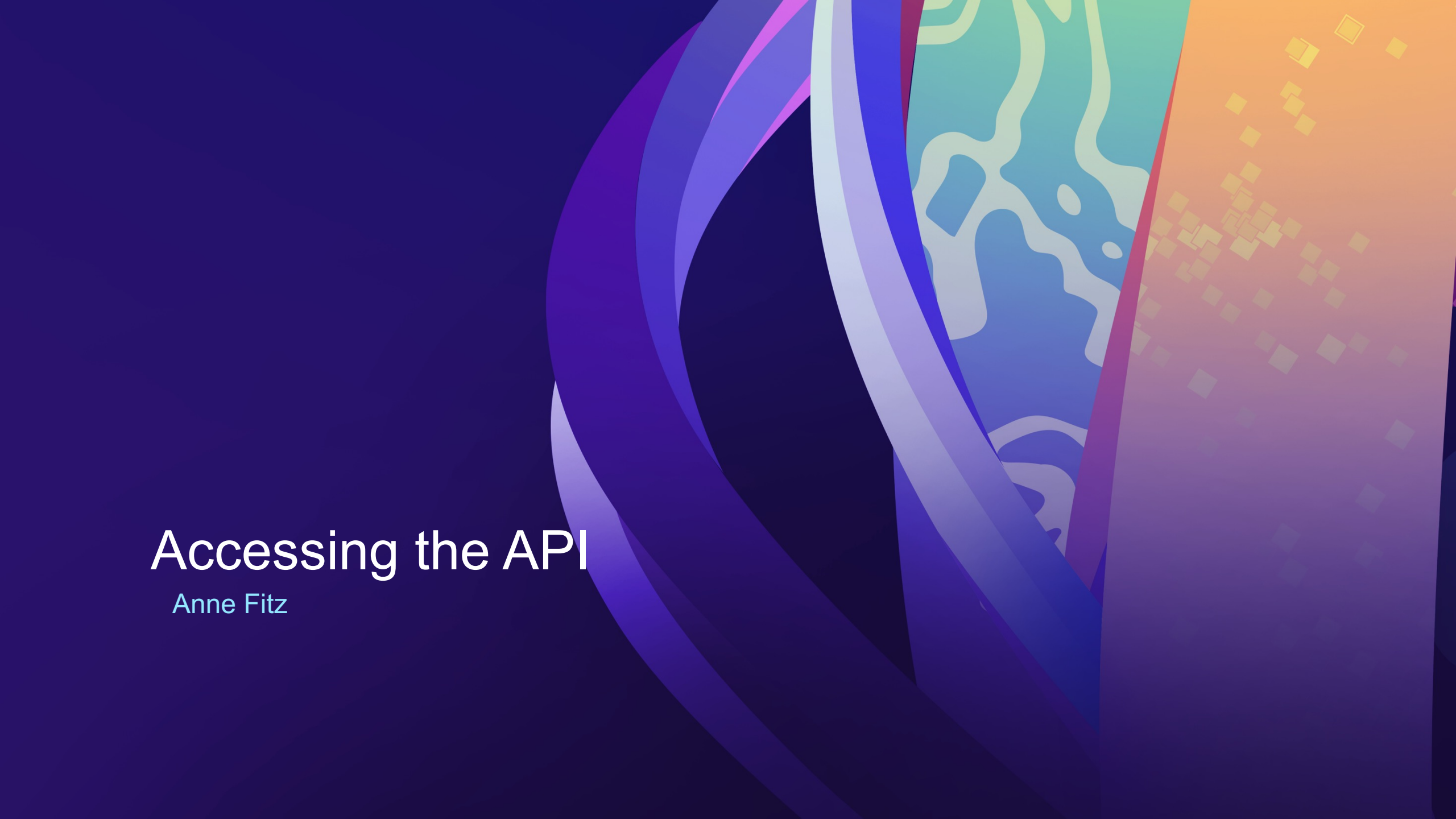
Start asking questions!



https://links.esri.com/UC22-programming-patterns

Agenda

- Accessing the API
- Maps and views
- Layers and visualization
- LayerViews and querying
- Event handling
- Programming patterns
- Widgets



Accessing the API

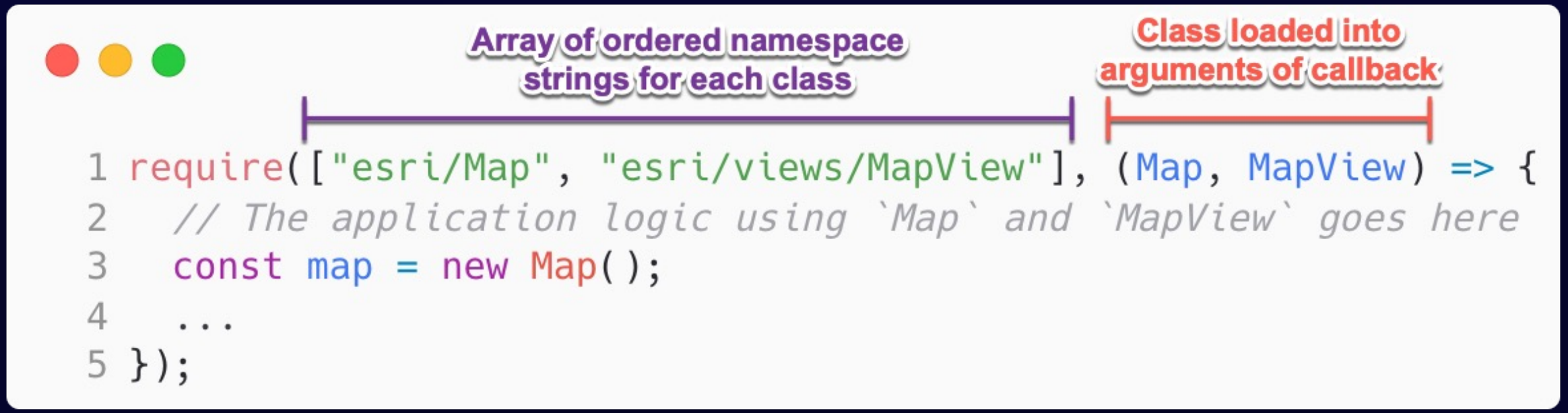
Anne Fitz

Asynchronous Module Definition (AMD)

AMD Modules via ArcGIS CDN

```
1 <link rel="stylesheet" href="https://js.arcgis.com/4.24/esri/themes/light/main.css">
2 <script src="https://js.arcgis.com/4.24/"></script>
```

Loading Classes via AMD



ECMAScript Modules (ESM)

ES modules via NPM

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

```
1 import Map from "@arcgis/core/Map";
2 import MapView from "@arcgis/core/views/MapView";
3
4 // Application logic using `Map` and `MapView` goes here
5 const map = new Map({
6   basemap: "topo-vector"
7 });
```

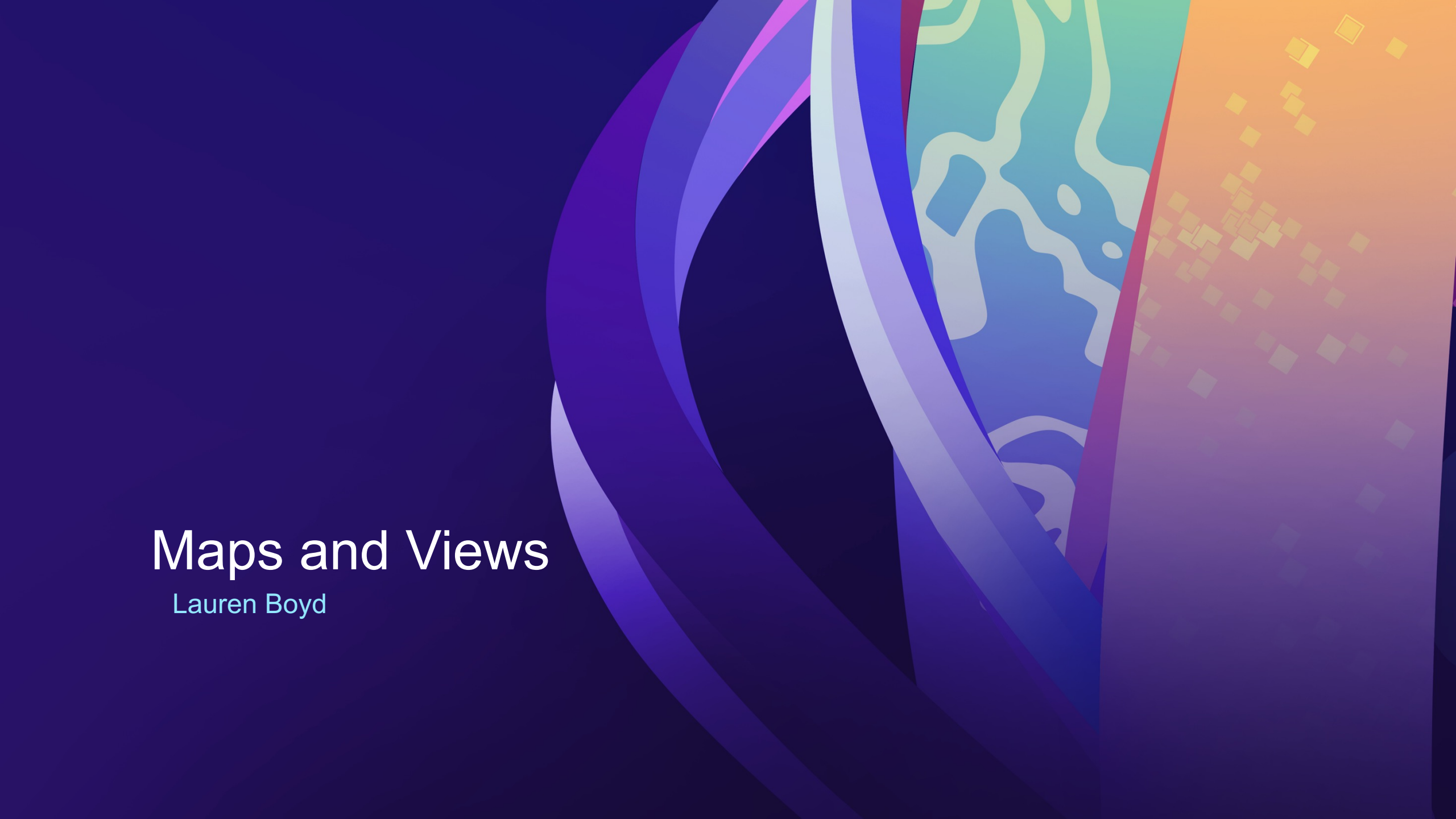
Constructors and setting properties

All classes have a constructor, property values can be set by passing parameters to the constructor

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

Properties can also be set directly on a class after it has been constructed

```
// update the basemap property
map.basemap = "gray-vector";
```



Maps and Views

Lauren Boyd

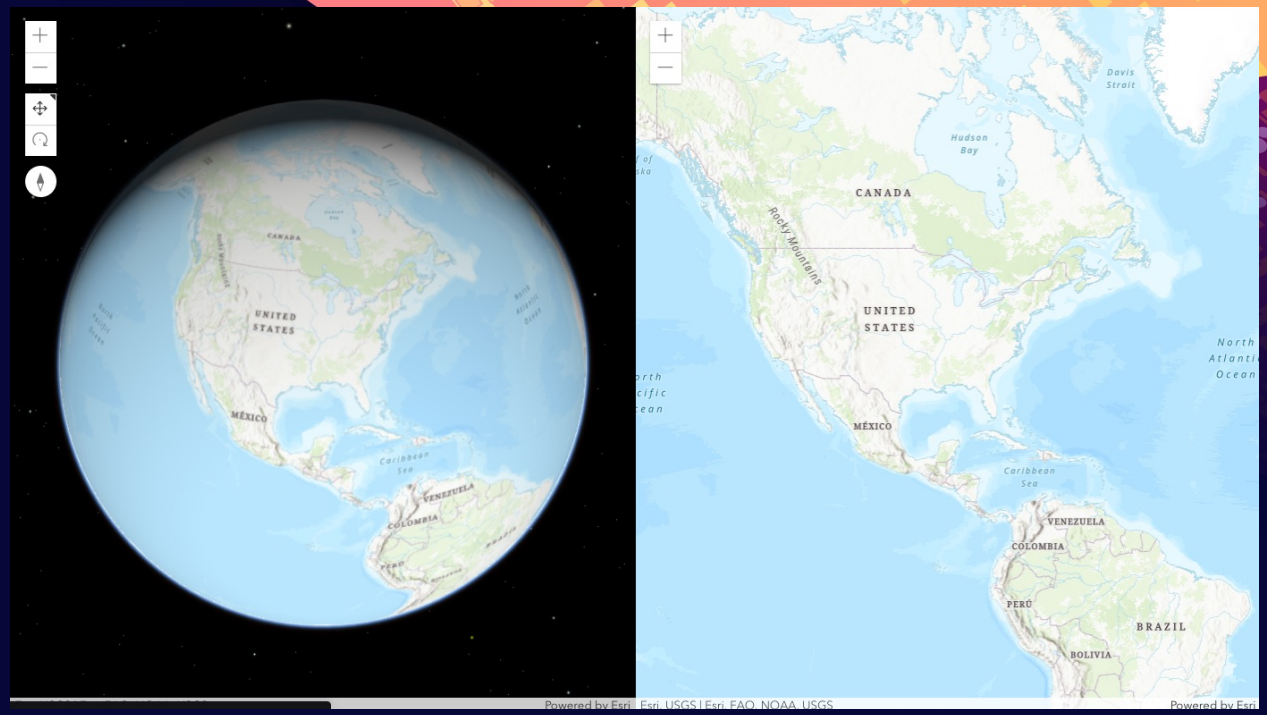
Map and Views

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

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

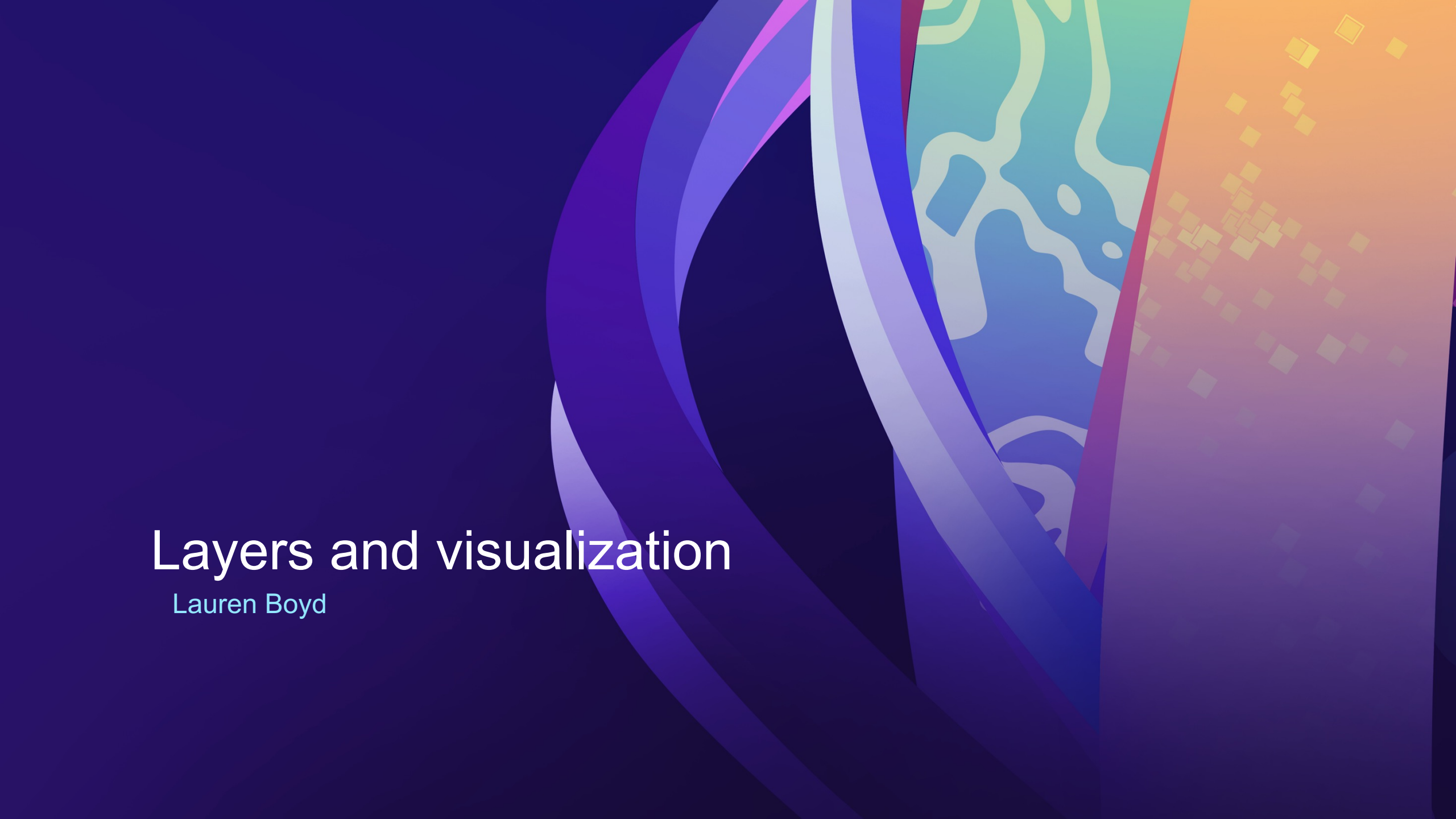
```
const map = new Map({
  /*
  For ArcGIS organizational usage:
   "satellite", "hybrid", "terrain", "oceans", "osm", "dark-gray-vector", "gray-vector", "streets-vector", "topo-vector", "streets-night-vector", "streets-relief-vector", "streets-navigation-vector"on-vector"
  For API key usage:
  "arcgis-imagery", "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"

   */
  basemap: "topo-vector"

  /*
   world-elevation
   */
  ground: "world-elevation"
})
```



Maps and Views

AMD 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, Heatmap, Opacity, Bloom, Aggregation, Thinning, Visible Scale Range
3D Visualization	Globes and local scenes, terrain renderering, cities in 3D, Visualizing points with 3D symbols

Renderers

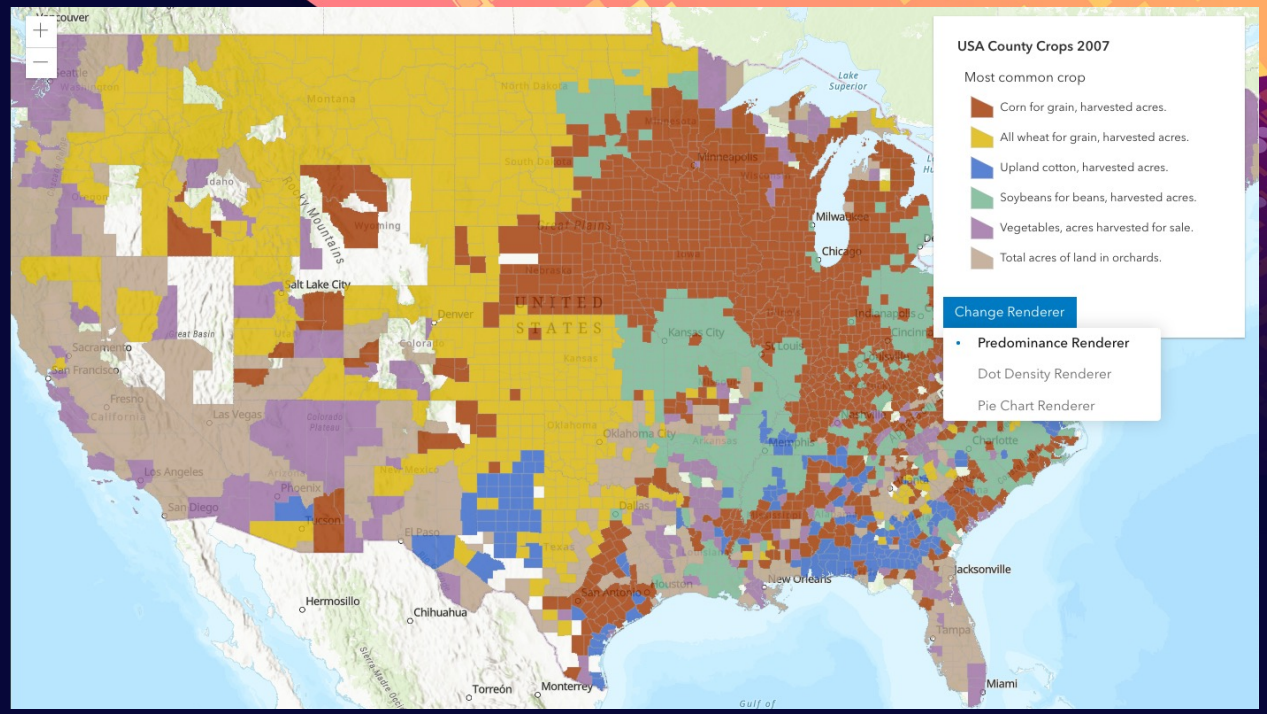
- Renderers define how to visually represent features in the layer

```
const povLayer = new FeatureLayer({
  url: "url to your service",
  renderer: {
    type: "simple", // autocasts as new SimpleRenderer()
    symbol: {
      // define symbol properties
    }
  }
});
```

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

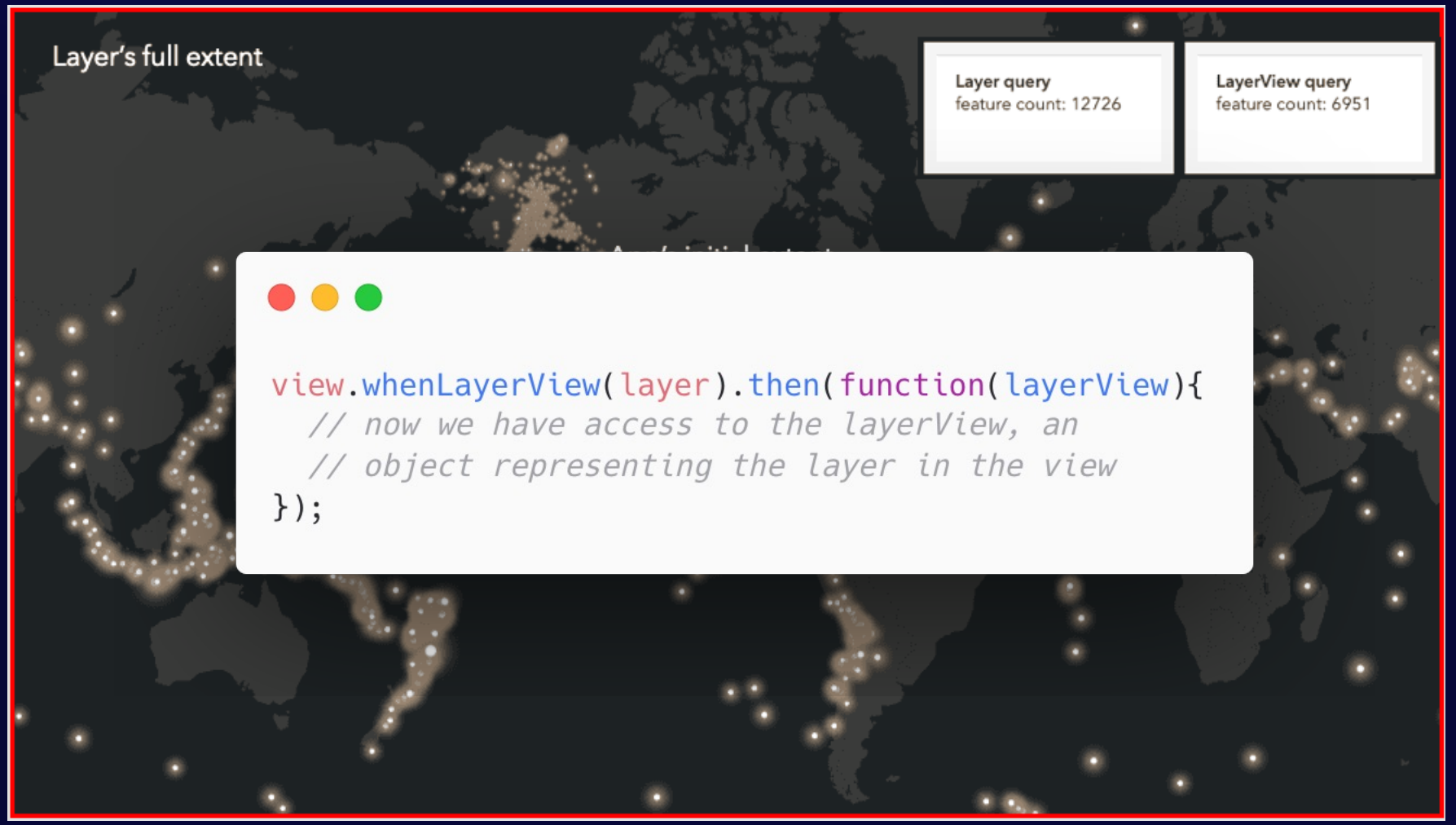
Demo

LayerViews and Querying

Anne Fitz

LayerViews

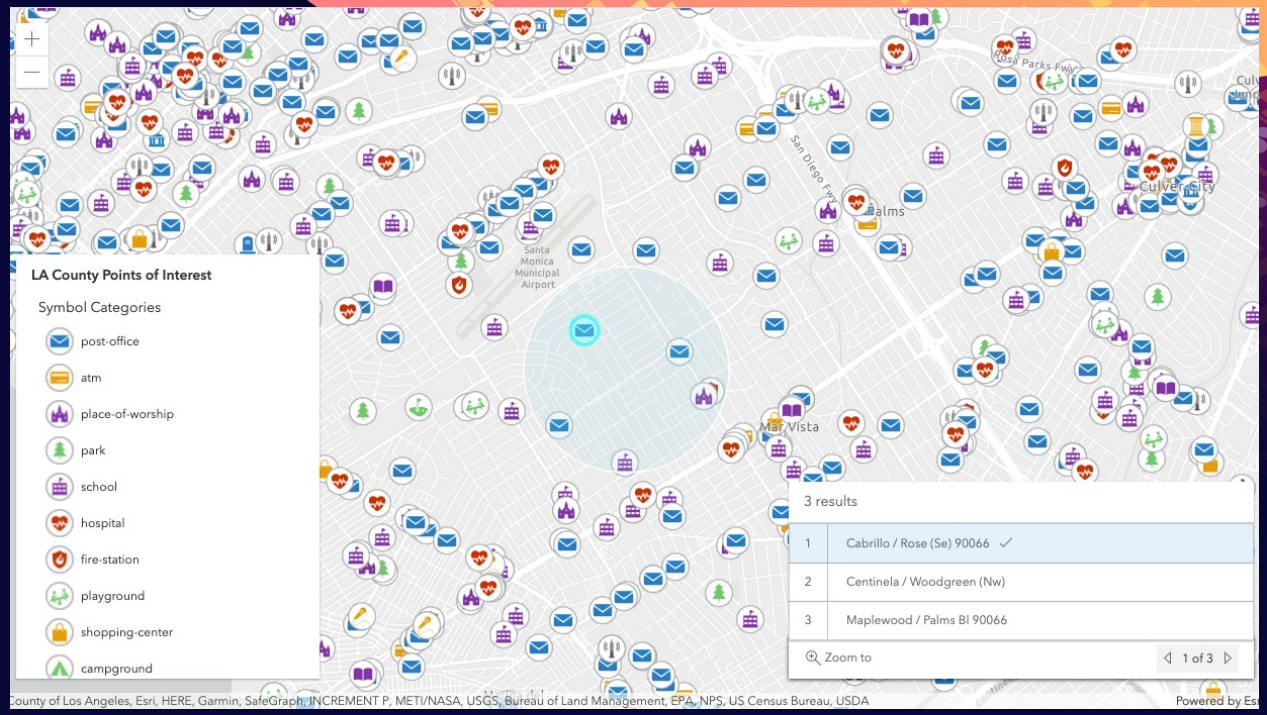
- Responsible for rendering features in the view
- Allows for client-side queries, filtering, effects, etc.



Queries

- Using layer.createQuery()
  - Query object already has the layers filters and layer definitions
  - more consistent
- Use new Query() when you don't want predefined filters to be applied

```
1 const query = featureLayer.createQuery();
2 query.where = "Field = '1'";
3 query.returnGeometry = true;
4 featureLayer.queryFeatures(query).then((results) ⇒ {
5   // do something with the results
6   console.log(results);
7 });
```



Querying

[Server-side]

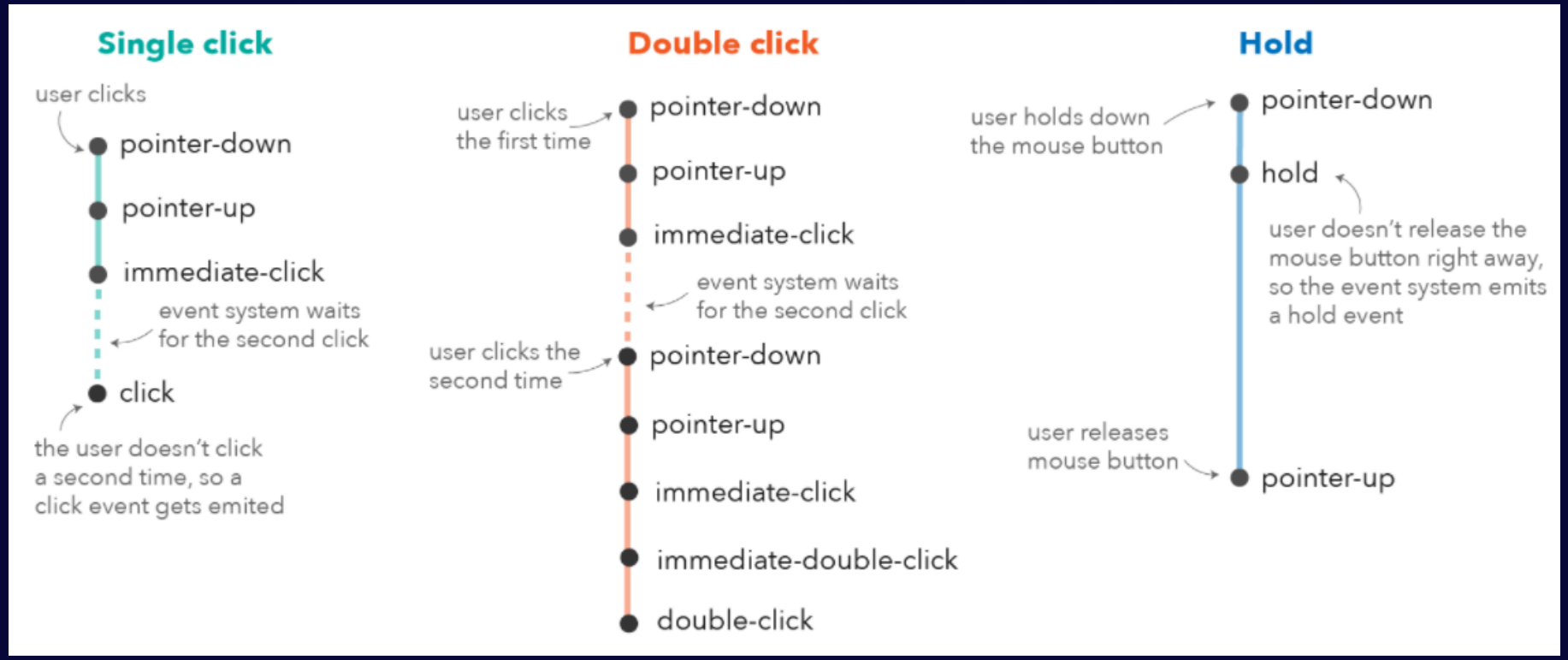
[Client-side]

Event handling

Anne Fitz

Events

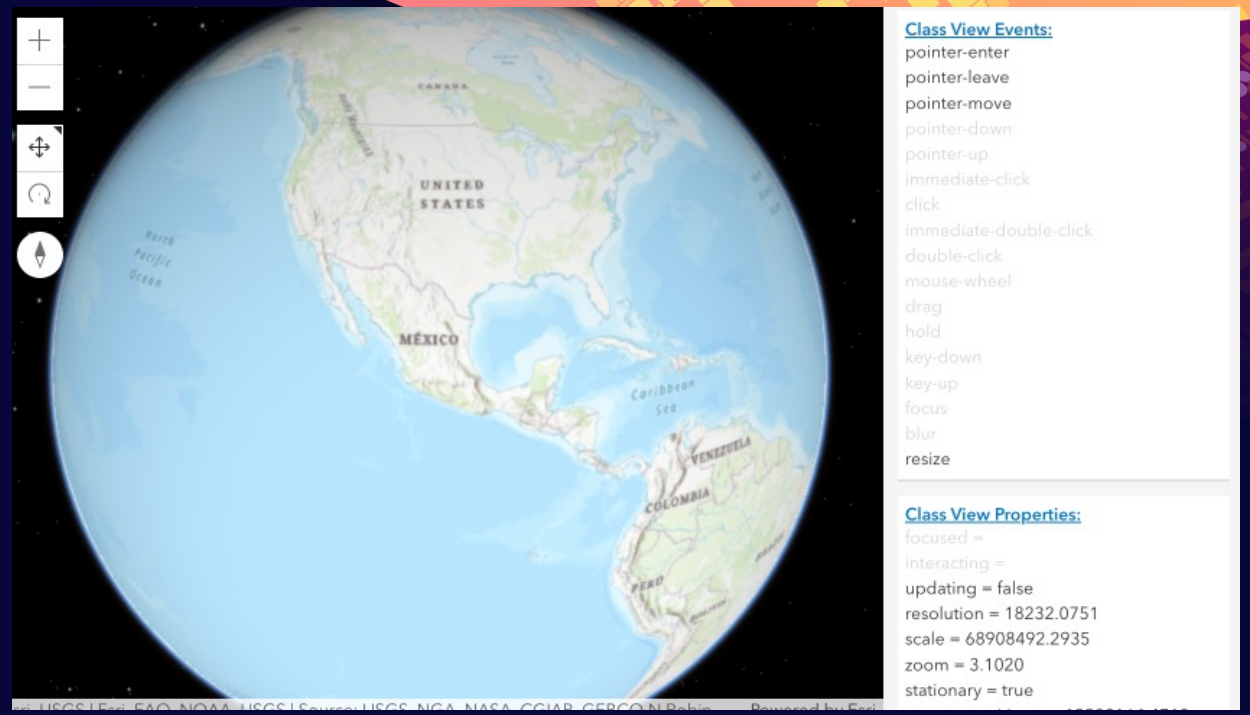
- List of Events



Event Handling

- Access features on pointer move

```
// get screenpoint from view's pointer-move event
view.on("pointer-move", (event)⇒{
  // Search for graphics on layers at the hovered location
  // exclude view.graphics from the hitTest
  view.hitTest(event, {exclude: view.graphics}).then((response) ⇒ {
    // if graphics are returned, do something with results
    const graphic = response.results[0].graphic;
  })
}
```



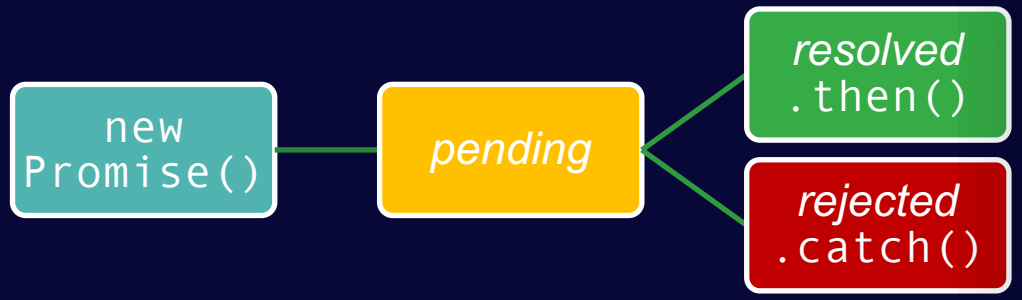
Event handling

Demo


Programming patterns
Lauren Boyd

Promises

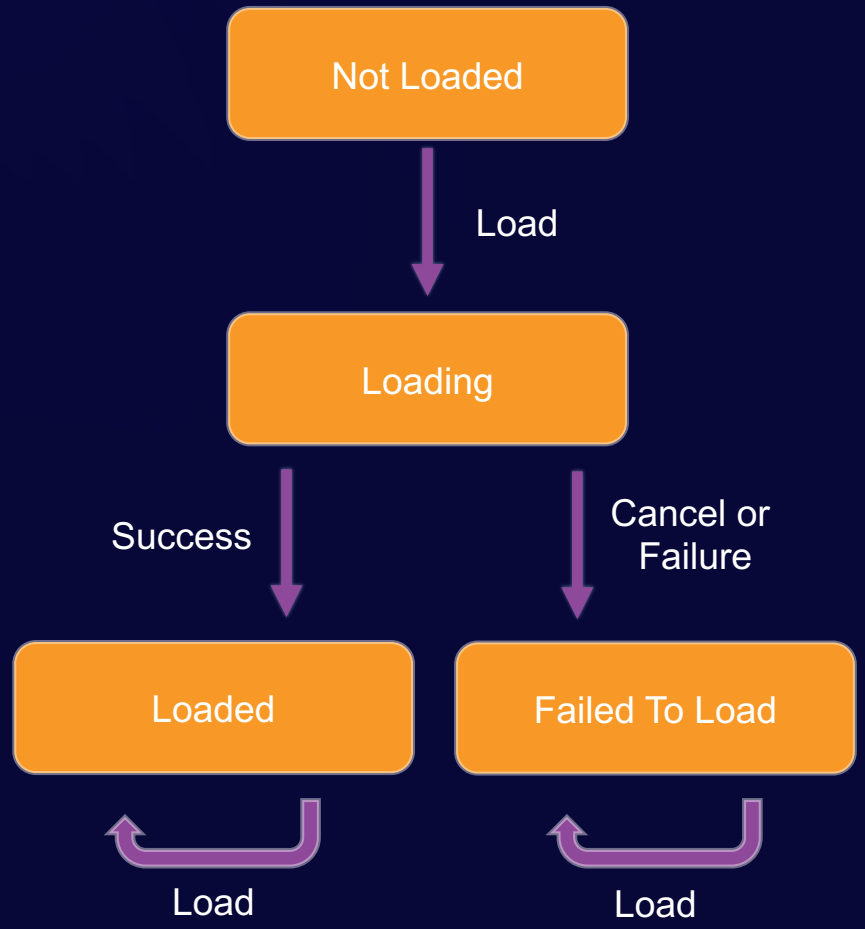
Promise = representation of a future value returned from an asynchronous task




```
view.goTo({
  center: [-126, 49]
})
.then(() => {
  // do something when view.goTo has
  //   completed successfully
})
.catch((error) => {
  // if view.goTo does not complete successfully,
  //   print the error to the console
  console.log(error);
});
```

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

```
// Reacting when a property changes
reactiveUtils.watch(
    ()=> view.map.basemap.title,
    (newValue, oldValue) => {
        console.log(`Old: ${oldValue}, New: ${newValue}`);
    });

// Call the callback initially as well as on any changes
reactiveUtils.watch(
    ()=> view.map.basemap.title,
    (newValue) => {
        console.log(`Basemap title: ${newValue}`);
    },
    { initial: true });

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

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}`);
    });
```

Widgets

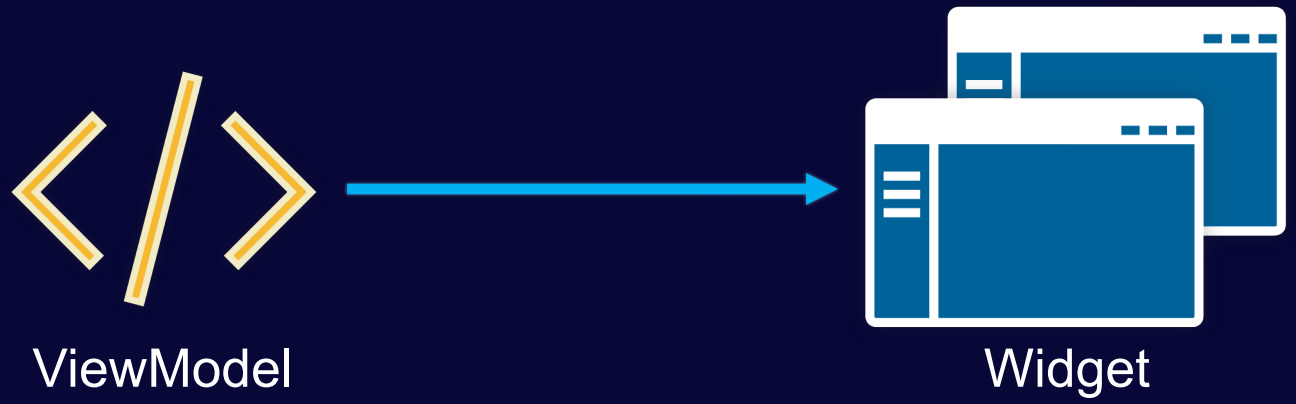
Lauren Boyd

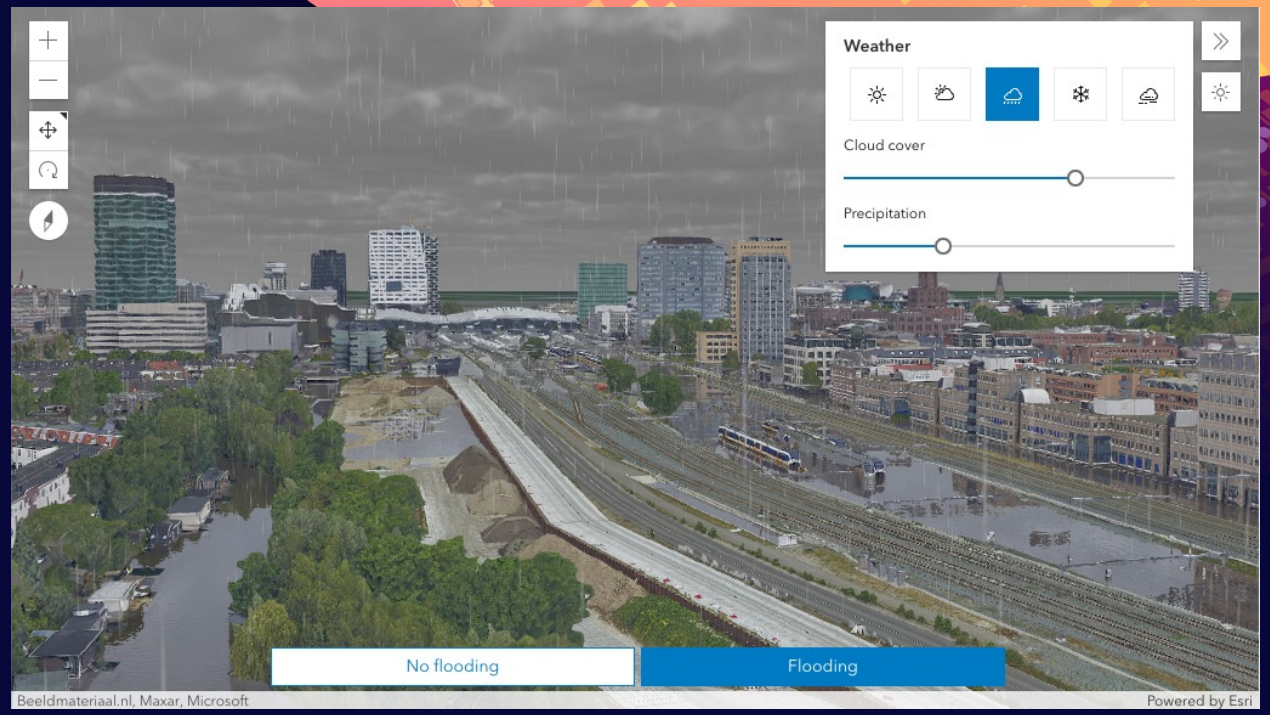
Widgets

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

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

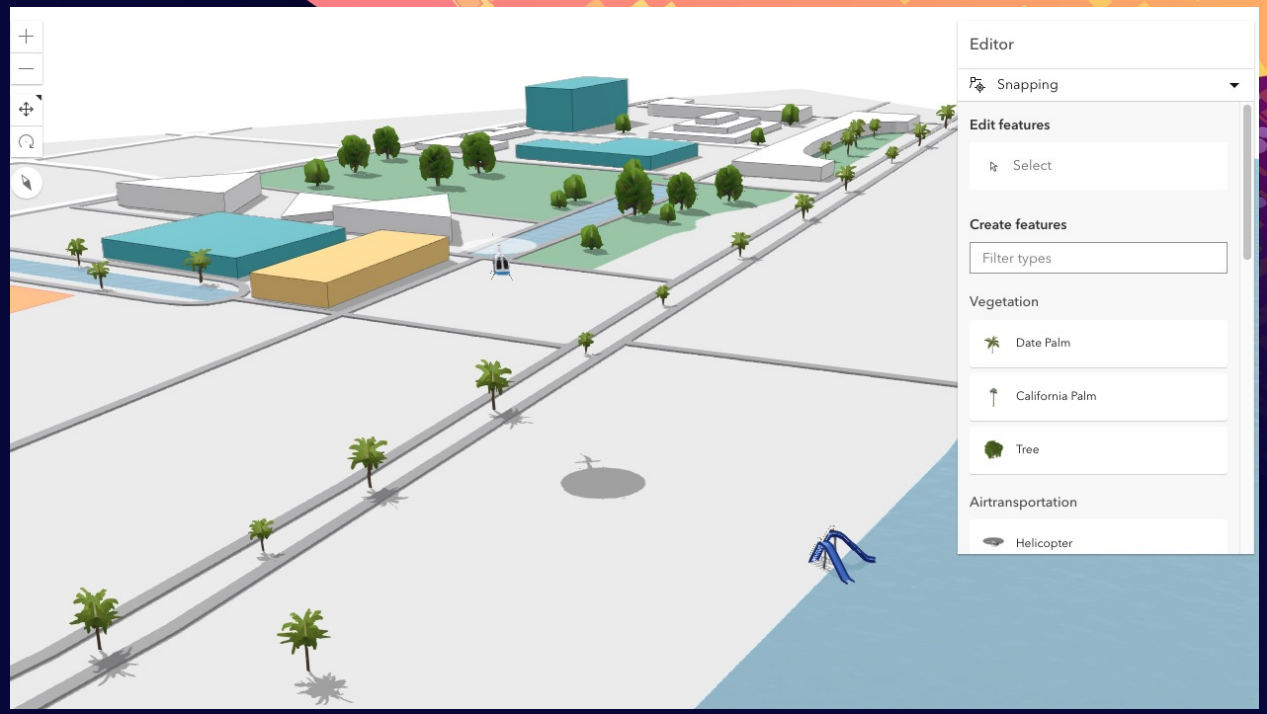
ViewModels





Weather

Demo


Editor
Snapping
Edit features
Select
Create features
Filter types
Vegetation
Date Palm
California Palm
Tree
Airtransportation
Helicopter

Editor

Demo

Where can I learn more?

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

Upcoming JS API Sessions

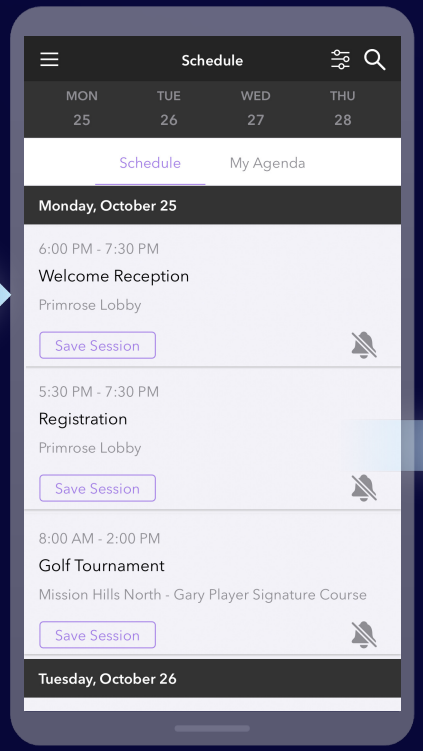
Session	Day/Time	Location
ArcGIS API for JavaScript: Web Editing	Wednesday, 1:00PM – 2:00PM	SDCC Room 10
	Thursday, 4:00PM – 5:00PM	SDCC Room 07 AB
Web Development: Strategies	Wednesday, 1:00PM – 2:00PM	SDCC Room 07 AB
ArcGIS API for JavaScript: Working with Imagery	Wednesday, 1:15PM – 2:00PM	SDCC Expo Demo Theater 12
ArcGIS API for JavaScript: An Introduction and What's New	Wednesday, 2:30PM – 3:30PM	SDCC Room 07 AB
ArcGIS API for JavaScript: The Road Ahead	Wednesday, 4:00PM – 5:00PM	SDCC Room 07 AB
ArcGIS API for JavaScript: Advanced Topics	Thursday, 10:00AM – 11:00AM	SDCC Room 16 A
ArcGIS API for JavaScript: Building 3D Web Apps	Thursday, 4:00PM – 5:00PM	SDCC Ballroom 6 E

https://links.esri.com/UC22-programming-patterns

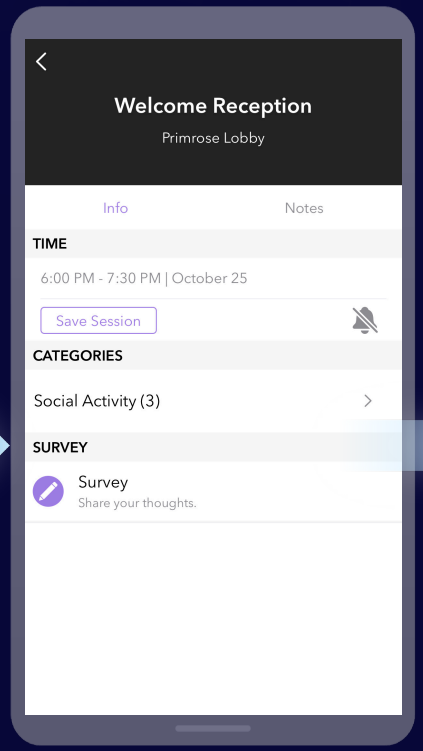
Please Share Your Feedback in the App

Download the Esri Events app and find your event

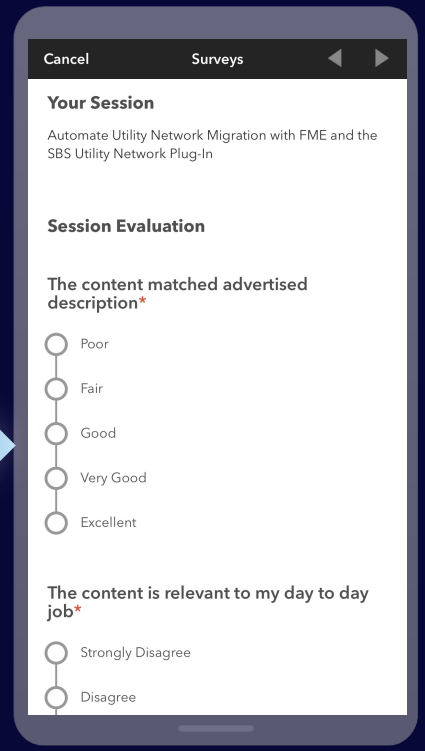
Select the session you attended



Scroll down to "Survey"



Log in to access the survey



Copyright © 2022 Esri. All rights reserved.