Getting Started with ArcGIS API for JavaScript

Kelly Hutchins
Nick Senger
Overview

- Setting up dev environment
- Debugging tips and tricks
- Tour of the SDK
- Build an app
  - Create map
  - Add layer
  - Make it interactive
  - Add widgets
• Focusing on version 4.0 for this presentation
• Concepts are similar between 3.x and 4.0
• Version 3.x UC 2015 presentation and demos are available

• TODO ADD LINK TO GITHUB REPO WITH SLIDES AND DEMOS
Where to begin?

Build 3D web apps!

The 4.x series of the ArcGIS API for JavaScript is Esri's next generation JavaScript API that integrates 2D and 3D into a single, easy-to-use, powerful API. Version 4.3 lets you build full-featured 3D applications powered by web scenes that can include rich information layers such as terrain, basemaps, imagery, features, integrated mesh layers, and 3D objects.

Learn about 4.3

https://developers.arcgis.com/javascript
## Choosing a version

<table>
<thead>
<tr>
<th>Capability</th>
<th>3.17</th>
<th>4.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D</td>
<td>Not available</td>
<td>Released</td>
</tr>
<tr>
<td>2D</td>
<td>Released</td>
<td>Released (partial support)</td>
</tr>
<tr>
<td>Vector Tile Layer</td>
<td>Released</td>
<td>Released</td>
</tr>
<tr>
<td>Raster Tile Layer</td>
<td>Released</td>
<td>Released</td>
</tr>
<tr>
<td>Imagery Layer</td>
<td>Released</td>
<td>Released</td>
</tr>
<tr>
<td>Map Image Layer</td>
<td>Released</td>
<td>Released (dynamic layers currently not supported)</td>
</tr>
<tr>
<td>Feature Layer</td>
<td>Released</td>
<td>Released (currently supports query and visualization)</td>
</tr>
<tr>
<td>Geometry Engine</td>
<td>Released</td>
<td>Released</td>
</tr>
<tr>
<td>Web Scene</td>
<td>Not available</td>
<td>Released</td>
</tr>
<tr>
<td>Web Map</td>
<td>Released</td>
<td>Released (partial support)</td>
</tr>
</tbody>
</table>

Developer Setup

- Choose an IDE
- Code assist
- Helpful tools
Get the API

- CDN
- Custom builds
- Download API

```html
// Reference the JavaScript API from our CDN and you are ready to get started:
<link rel="stylesheet" href="https://js.arcgis.com/4.3/esri/css/main.css">
<script src="https://js.arcgis.com/4.3/"></script>
```

CSS

- Main.css
- View.css
- Custom css (SASS)

```html
<link rel="stylesheet" href="https://js.arcgis.com/4.3/esri/css/main.css">
<link rel="stylesheet" href="https://js.arcgis.com/4.3/esri/css/view.css">
```

JSAPI Resources

- Collection of resources to aid development in the JSAPI
- Some of these include:
  - **JSHint file**
    - Helps to detect errors and potential problems in code
  - **Typescript definition file**
    - Provides code assist in some IDE’s such as Visual Studio Code
  - Build tools, e.g. **Bower** for custom builds of the API
  - Callback page for popups using OAuth2 for security

https://github.com/Esri/jsapi-resources
Build first app
First Steps

• Plan application structure
  - Demo structure
    - Separate css, html, js
    - Reference JSAPI, CSS etc.

```html
<title>Step 1: Create a map</title>
<link rel="stylesheet" href="https://js.arcgis.com/4.3/esri/css/main.css">
<link rel="stylesheet" href="css/main.css">
<script src="https://js.arcgis.com/4.3/"></script>
<script src="js/main.js"></script>
```
Make a map

```javascript
var map = new Map({
  basemap: "streets"
});
```

# Debugging Tips

## Module Order

- Order matters
- TypeScript helps

```javascript
Uncaught TypeError: Cannot read property 'move' of undefined
  at main.js:28
  at ca (init.js:18)
  at init.js:18
  at G (init.js:18)
  at ka (init.js:18)
  at v (init.js:20)
  at HTMLScriptElement.<anonymous> (init.js:23)
```

## Missing CSS

- Unstyled widgets
- Scrambled tiles
Map View

```javascript
view = new MapView({
  container: "viewDiv",
  map: map,
  zoom: 12,
  center: [-117.168, 32.776]
});
```

Demo

Get started with map
Add data to map
Add data

- Many layer types
  - Feature layer
  - Map image layer
  - Tile Layer
  - Vector Tile Layer
  - .... and more
Layer coding pattern

1. Load module
2. Create layer
3. Set properties
4. Add to map (or scene)

```javascript
require(["esri/layers/FeatureLayer"], function(FeatureLayer){
  // points to the states layer in a service storing U.S. census
  var fl = new FeatureLayer({
    url: "https://sampleserver6.arcgisonline.com/arcgis/rest/services/Geography/States/FeatureServer/0"});
  map.add(fl); // adds the layer to the map
});
```
Working with properties

• Set properties in constructor

```javascript
var map = new Map({
    basemap: "topo",
    layers: [hoods, poi, beaches]
});
```

• Or directly on the instance

```javascript
view.center = [ -100, 40 ];
view.zoom = 6;
map.basemap = 'oceans';
```

Working with properties – continued

- Watch for property changes

  ```javascript
  layer.watch("loadStatus", function(status) {/**/});
  ```

- Use esri/core/watchUtils to aid in watching property changes
  - Once
  - When false
  - When true

Demo

Add layer to app
Symbolize data
Symbols

- Define symbols used to represent your data
- Many types
  - SimpleMarkerSymbol
  - PictureMarkerSymbol
  - SimpleLineSymbol
  - TextSymbol
  - ..... and more

Renderers

- **Define** a set of symbols and rules for drawing the layer
- Apply renderer to the layer

```javascript
var citiesRenderer = new SimpleRenderer({
  symbol: new SimpleMarkerSymbol({
    size: 10,
    color: "#FF4000",
    outline: {} // autcasts as new SimpleLineSymbol()
      color: [255, 64, 0, 0.4], // autcasts as new Color()
    width: 7
  }},
  label: "Major cities" // this will appear next to the symbol in the legend
});
```
Autocasting

- Use a property without having to import module
- Look for autocast label

Autocast code sample

```javascript
var sym = SimpleMarkerSymbol({
  color: "red",
  outline: {
    color: [128, 128, 128, 0.5],
    width: "0.5px"
  }
});
```
Demo

Apply renderer to layer
Make app interactive
Add interactivity: Popups

- Responsive
- Details about
  - Clicked location
  - Feature
  - Search results
- Customizable

Popup components

- All views have default popup
- Define title and content
  - Popup
  - Popup Template
- Custom actions
Popup Template

- Define content in PopupTemplate
- Associate template with layer

```javascript
var beaches = new FeatureLayer({
    url: "<URL to featurelayer>",
    // Step 4: Specify the outfields
    outFields: ["*"],
    popupTemplate: popupTemplate,
    // set renderer
    renderer: beachRenderer
});
```

// Create the popuptemplate
var popupTemplate = new PopupTemplate({
    title: "<b>Beach: {NAME}</b>",
    // Specify the content, first set the display fields
    content: [{
        type: "fields",
        fieldInfos: [{
            fieldName: "ADDRESS",
            visible: true,
            label: "Address: "
        },
```
Dockable

- Define dock position
  - bottom-right
  - top-right
  - bottom-left
  - top-left
  - bottom-center
  - top-center

```javascript
popup: {
  dockEnabled: true,
  dockOptions: {
    buttonEnabled: true,
    position: "bottom-right"
  }
}
```

Popup: Custom Actions

Trail run

Morton Peak - great run. great trail

🔍 Zoom to 🗑️ Measure Length
Demo

Add popups
Web maps
WebMap

- Write less code!
- Use ArcGIS Online/Portal Map Viewer to:
  - Specify basemap and extent
  - Define symbology
  - Create popup content
  - Add layers

```javascript
var map = new WebMap(
  portalItem: {
    id: "12843e71a82446d89a1df878312d6749"
  }
);
```

Demo
Use WebMap
Widgets
Widgets

- Basemap
- Compass
- Home
- Legend
- Search
- Expand
- Locate
- .... and more

Widget coding pattern

- Load module
- Create widget
- Set properties
- Add to View’s UI

```javascript
view.then(function() {
    var beaches = map.layers.getItemAt(1);
    // Step 1: Create the widget
    var legend = new Legend({
        // Step 2: Specify any additional properties
        // we are just setting the view to
        view: view,
        layerInfos: [{
            layer: beaches,
            title: "San Diego beaches"
        }]
    });
    // Step 3: Add the widget to the view's
    view.ui.add(legend, "bottom-left");
});
```
View UI

- Simple process of placing widgets in app

Customize widget appearance

- CSS
- SASS

Demo
Add widget to app
Which Font is Best For Your Presentation?

Avenir Next
If you are the sole presenter and will not distribute your PowerPoint presentation to others outside of Esri and will only be using your own Esri computer for presenting, please use the new font Avenir Next LT Pro.

See instructions on the following slide to see how to use this new font.

Arial
If you will be distributing your PowerPoint presentation to others outside of Esri, or using various computers to present please use Arial.

Using Arial gives your PowerPoint presentation the flexibility to be used by many people on various machines without the risk of font compatibility and changing the look of your layout.

Use Avenir Next if …
Sole Presenter
Single computer used for presentation

Use Arial if…
Multiple Users / Presentation Distributed to others
Presentation used on multiple computers
How to Change Your Theme Fonts to Avenir Next

Windows
Select: Design
  Variants (arrow)
  Fonts > Esri-Avenir Next

Mac
Select: Design
  Variants (arrow)
  Fonts > Esri-Avenir Next

What if my fonts don’t change?

If the text boxes on your slides are not using (Heading) or (Body) fonts from the Theme Fonts section, they will not update correctly. You will have to replace them manually.
How to Test if you’re using Avenir Next

If you have Avenir Next installed, the two Qs will match.
Did you know?

To quickly update any current presentation
Copy and paste your slides from your current deck into the newest template. The title and ending logo slide will update automatically.

For More Templates, Sample Files, and Icons See
https://compass.esri.com/resources/presentations/Pages/Main.aspx

Videos not working?
For video in PowerPoint, animated GIF is recommended. AVI and WMV (native PC files) will work best on a PC platform. H.264 MP4 will work only on PowerPoint 2013 and later.
If this shape does not appear as a perfect circle, adjust the aspect ratio of your display until it does.

Try the resolution 1920x1080 for 16:9 displays.