ArcGIS API for JavaScript: Building Apps with React

Tom Wayson René Rubalcava

@tomwayson @odoenet

2021 ESRI DEVELOPER SUMMIT

Good News

This is easier than ever!

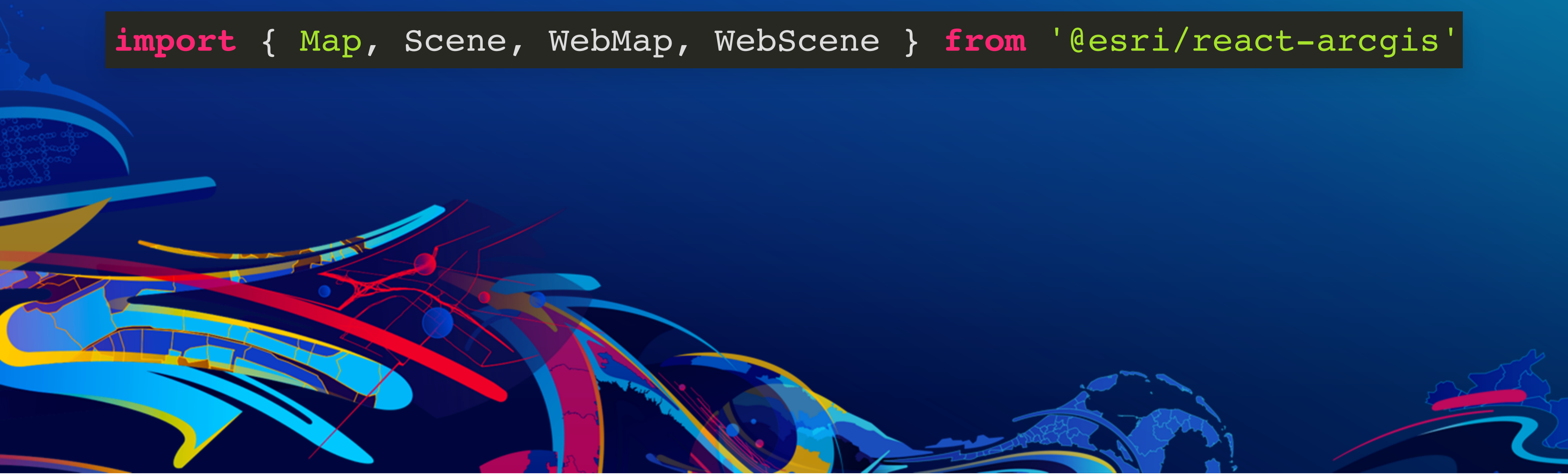
It just works

Tom, Dec 2020

@esri/react-arcgis

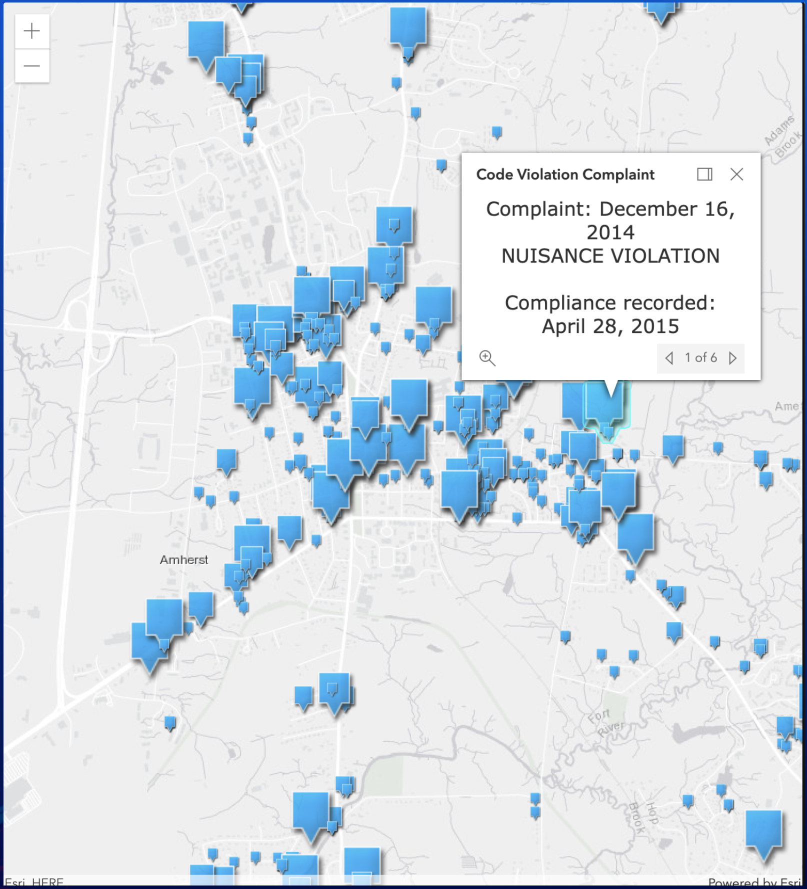
Ready to use components

```
import { Map, Scene, WebMap, WebScene } from '@esri/react-arcgis'
```



Web Map Component

```
<WebMap id="6627e1dd5f594160ac60f9dfc411673f" />
```


Code Violation Complaint

Complaint: December 16, 2014
NUISANCE VIOLATION

Compliance recorded: April 28, 2015

Web Scene Component

```
<WebScene id="0614ea1f9dd043e9ba157b9c20d3c538" />
```

Maps and Views

```
const mapProps = { basemap: "topo" };
const viewProps = {
    center: [-122.4443, 47.2529],
    zoom: 6
};
```

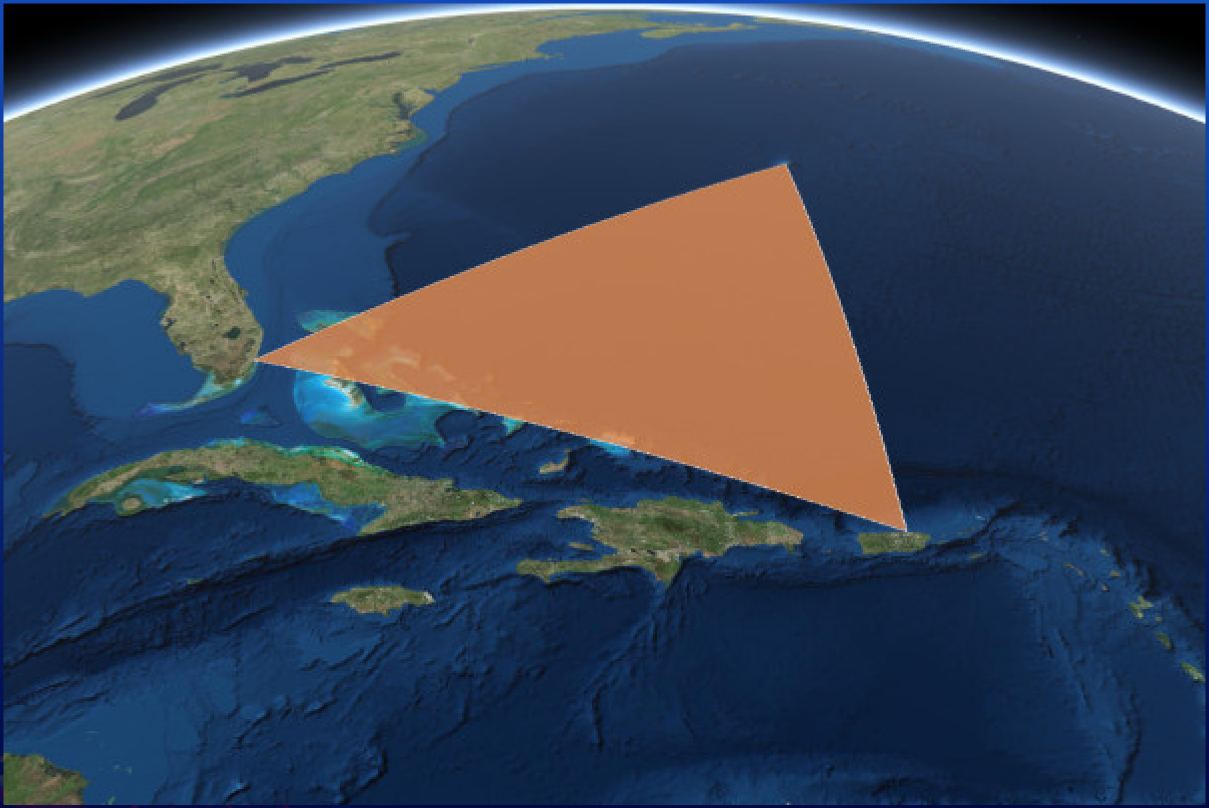
```
<Map mapProperties={mapProps} viewProperties={viewProps} />
```

or

```
<Scene mapProperties={mapProps} viewProperties={viewProps}  />
```

Custom Components

```
<Scene class="full-screen-map">
  <BermudaTriangle />
</Scene>
```



Creating Your Own Components

Creating a Map View

1. container (`<div id="map"></div>`)
2. map properties (`basemap: 'topo-vector'`)
3. view properties (`zoom: 8`)

Creating a Map View (no React)

```html
<div id="map"></div>
```

```javascript
const map = new ArcGISMap({
  basemap: 'topo-vector'
});
const view = new MapView({
  container: document.getElementById('map')
  map,
  center: [-118, 34],
  zoom: 8
});
```

Creating a Map View in React

1. Component renders container node
2. Create map view as a side effect

Component Concepts

- rendering
- refs
- side effects
- props/state

React hooks

- `useRef()`
- `useEffect()`
- `useState()`

See the Using Webpack and React (2020) slides for class based examples

Rendering the container

```
import React from 'react';

export const MapView = () => {
  return <div id="map" />;
};
```

Rendering the container

```
import React from 'react';

export const MapView = () => {
  return <div id="map" />;
};
```

Anti-pattern: do **not** render `id` in component

useRef for the container

```
import React, { useRef } from 'react';

export const MapView = () => {
  const mapRef = useRef();

  return <div ref={mapRef} />;
};
```

`useEffect` for side effects

```
import React, { useRef, useEffect } from 'react';

export const MapView = () => {
  const mapRef = useRef();
  useEffect(
    () => { /* what you want to do */ },
    [] /* when you want to do it */
  });
  return <div ref={mapRef} />;
};
```

`useEffect` to create map & view

```
const mapRef = useRef();
useEffect(
  () => {
    // create map and view
    const map = new ArcGISMap({
      basemap: 'topo-vector'
    });
    const view = new MapView({
      container: mapRef.current,
      map: map,
      center: [-118, 34],
      zoom: 8
    });
  }
, []); // only after initial render
```

<MapView /> Component

```
import React, { useRef, useEffect } from 'react';
import createMapView from './utils/map';

export const MapView = () => {
  const mapRef = useRef();
  useEffect(() => {
    // create map and view
    const view = createMapView(mapRef.current);
    // clean up
    return () => { view && view.destroy(); };
  }, []); // only after initial render
  return <div ref={mapRef} />;
}
```

🎉 Success! 🎉

✅ created a component that renders a `container`

✅ created a map view after the initial render

✅ only destroy `MapView` when unmounting

Map & view properties 🤔

```
<MapView basemap="streets" zoom="13" />
```

Use React props

```
export const MapView = ({ basemap, zoom }) => {
  const mapRef = useRef();
  useEffect(() => {
    // read map and view properties from props
    const mapProperties = { basemap };
    const viewProperties = { zoom };
    // create map and view
    const view = createMapView(mapRef.current, mapProperties, vie
    // clean up
    return () => { view && view.destroy(); };
  }, []); // only after initial render
  return <div ref={mapRef} />;
}
```

Update map & view properties

```
<MapView basemap={{basemap}} zoom="13" />
<BasemapSelect value={{basemap}} onChange={{setBasemap}} />
```

useState

```
const [basemap, setBasemap] = useState('topo-vector');
```

useState

```
export const MapPage => () {
  const [basemap, setBasemap] = useState('topo-vector');
  return (
    <MapView basemap={{basemap}} zoom="13" />
    <BasemapSelect value={{basemap}} onChange={{setBasemap}} />
  );
}
```

Use another effect in `<MapView>`

```
useEffect(() => {
  // TODO: view is undefined
  view.map.basemap = basemap;
}, [basemap]); // called whenever basemap prop changes
```

Hold onto view in state

```
// in MapView component
const [view, setView] = useState(null);
// later in useEffect()
setView(createMapView(mapRef.current, mapProperties, viewProperti
```

Only update if view has been created

```
useEffect(() => {
  if (!view) {
    // this was called before setView()
    return;
  }
  view.map.basemap = basemap;
}, [view, basemap]);
```

🎉 Success! 🎉

✅ initialize map & view properties from `props`

✅ update map or view when `props` change

Watch or event callbacks 🤔

```
<MapView basemap={{basemap}} zoom="13" onClick={{logClick}} />
```

Wire up handlers in another effect

```
useEffect(() => {
  if (!view) {
    return;
  }
  const handle = view.on('click', onClick);
  return function removeHandle() {
    handle && handle.remove();
  };
}, [view, onClick]);
```

use clean-up functions to remove event & watch handlers

Components

A bridge between your React app and the ArcGIS API

Modern React and the ArcGIS API

Manage global state in React

- You may not need Redux/MobX
- Context is powerful, and injectable

useContext hook

```
import ThemeContext from './ThemeContext';

const ThemedMap = () => {
  const theme = useContext(ThemeContext);
  const basemap = theme === 'dark'
    ? 'dark-gray'
    : 'gray';
  return (
    <Map basemap={basemap} />
  );
};
```

Modularize API usage

- Do all the API work separate from your UI
- *Separate content from navigation* - pattern in PWAs
- Mock/stub API in tests

```
// src/data/map.ts
export function initialize(element: Element) {
  view.container = element;
  view.when(() => {
    // magic
  });
}
```

- Use in your context or component

```
const elRef = useRef(null);
useEffect(
  () => {
    const loadMap = async () => {
      const map = await import("../data/map");
      map.initialize(elRef.current);
    };
    loadMap();
  },
  []
);
```

Why lazy load the API?

- So webpack can create async bundles
- `bundle1.js` -> `bundle2.js` -> `bundle3.js`
- Only load the resources you need when you need them
- Leads to faster initial loads

Suspense

Hold your Suspense

- Lazy-load entire React components
- useful in modular apps

```
import React, { lazy, Suspense } from "react";
// lazy load the components that use Maps
const WebMapView = lazy(() => import("../components/WebMapView"))
// later on
<Suspense  fallback={<div>Loading...</div>}>
  <WebMapView />
</Suspense>
```

Hold your Suspense

- Still not out of beta, so use at your own risk

Demo: React with ESM

@arcgis/core

🌐 ❤ 📦 R

ArcGIS API is different

- powerful library with large footprint
- uses dynamic module loading & web workers

ArcGIS API is different

- powerful library with large footprint
- uses dynamic module loading & web workers
- can slow your build; or not work w/ defaults

Is your bundler smarter than you?

🌐 🤔 📦 ⛰

Try esri-loader

Installing esri-loader

```
npm install --save esri-loader
```

Installing esri-loader

```
yarn add esri-loader
```

Using `loadModules()`

```
import { loadModules } from 'esri-loader';

loadModules([
  "esri/Map",
  "esri/views/MapView"
]).then(([Map, MapView]) => {
  // Code to create the map and view will go here
});
```

How it works

```
// calls require() once the ArcGIS script is loaded

require([
  "esri/Map",
  "esri/views/MapView"
], (Map, MapView) => {
  // Code to create the map and view will go here
});
```

Lazy loads the ArcGIS API

```
 // injects a script tag the first time
const esriConfig = await loadModules(["esri/config"])
esriConfig.useIdentity = false;

// don't worry, this won't load the API again!
const [Map, MapView] = await loadModules(
  ["esri/Map", "esri/views/MapView"]
);
```

Lazy loads the ArcGIS API

```
 // injects a script tag the first time
const esriConfig = await loadModules(["esri/config"])
esriConfig.useIdentity = false;

// don't worry, this won't load the API again!
const [Map, MapView] = await loadModules(
  ["esri/Map", "esri/views/MapView"]
);
```

Defaults to latest CDN version

esri-loader options

- Use an earlier release, even 3.x!
- Use a local AMD build
- Lazy load CSS

Keeps ArcGIS API out of your build

Keeps ArcGIS API out of your build

- faster builds
- greater tool compatibility

@esri/react-arcgis uses esri-loader

```
npm i --save esri-loader @esri/react-arcgis
```

esri-loader-hooks

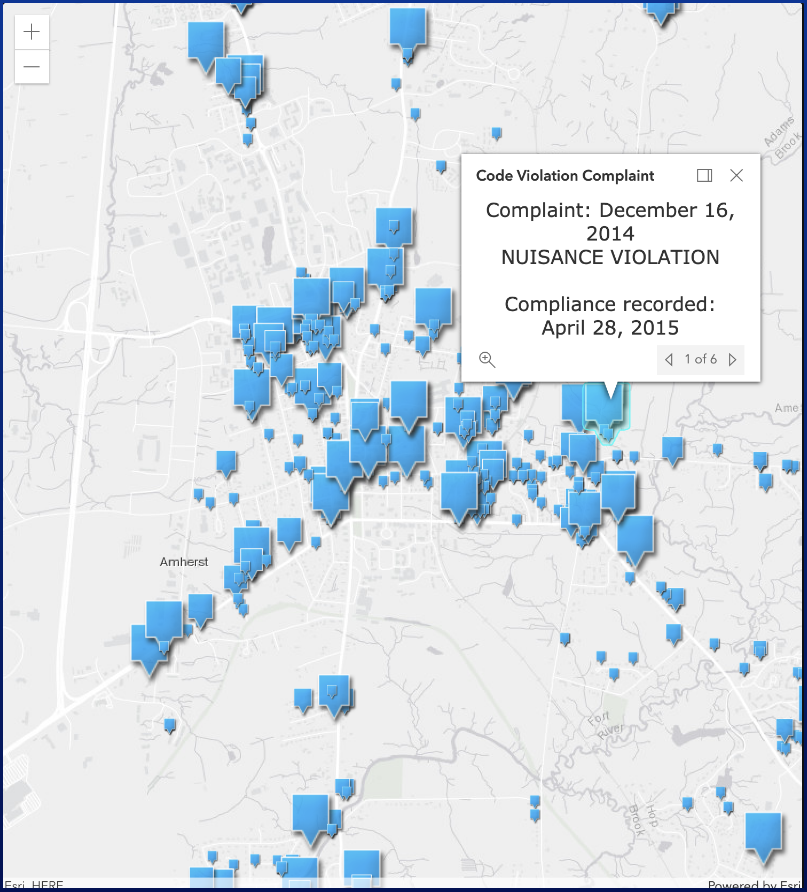
```
npm i --save esri-loader esri-loader-hooks
```

esri-loader-hooks

```
import {
  useMap, useScene, useWebMap, useWebScene, // create a map or sc
  useEvent, useEvents, useWatch, useWatches, // handle events or
  useGraphic, useGraphics // add graphics to a map/scene
} from 'esri-loader-hooks';
```

esri-loader-hooks

```
function WebMap() {
  const [ref] = useWebMap('6627e1dd5f594160ac60f9dfc411673f');
  return <div style={{ height: 400 }} ref={ref} />;
}
```

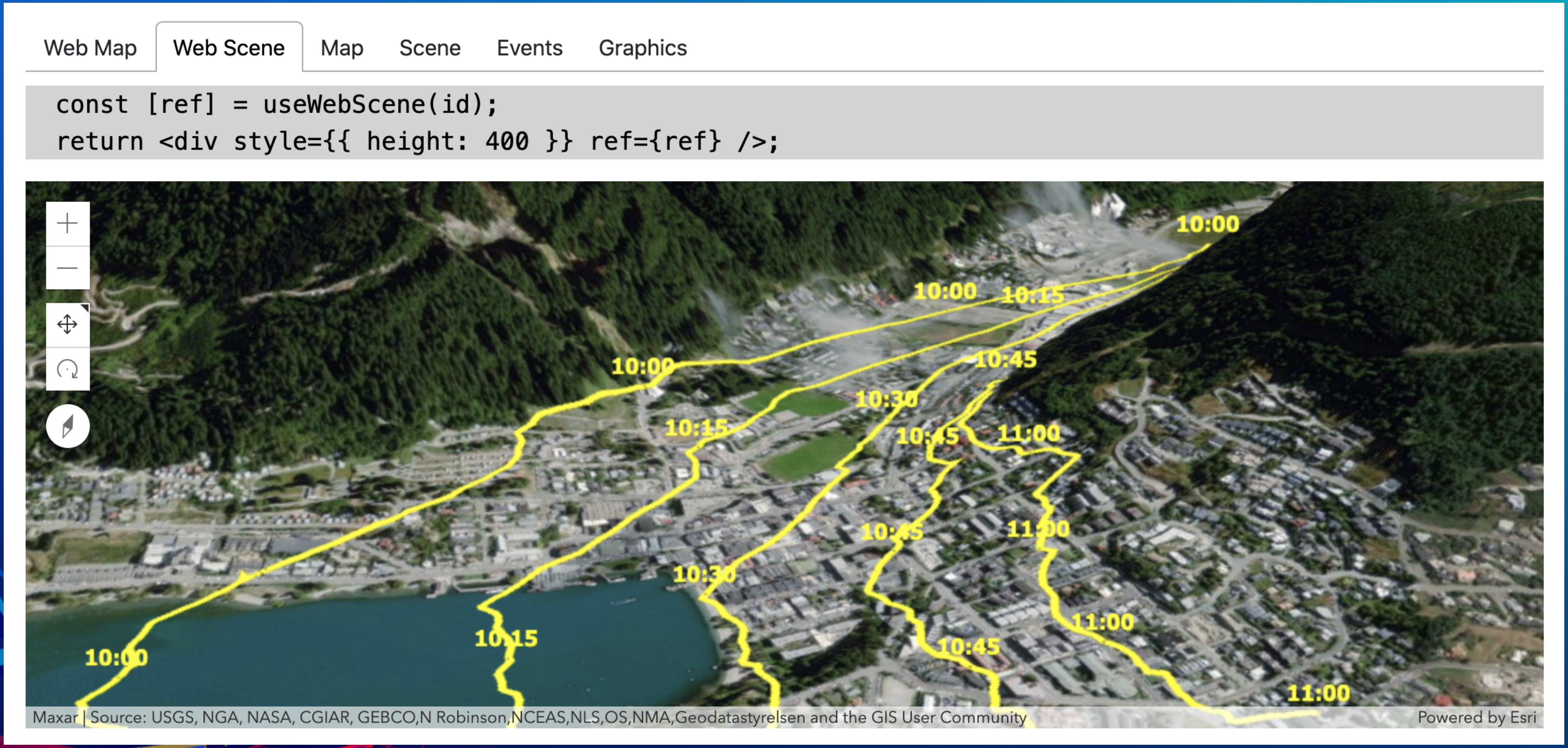

Code Violation Complaint

Complaint: December 16, 2014
NUISANCE VIOLATION

Compliance recorded: April 28, 2015

1 of 6

Example: esri-loader-hooks



```
const [ref] = useWebScene(id);
return <div style={{ height: 400 }} ref={ref} />;
```

10:00
10:00
10:15
10:00
10:45
10:30
10:15
10:45
11:00
10:45
11:00
10:30
11:00
10:15
10:45
10:00
11:00
Maxar | Source: USGS, NGA, NASA, CGIAR, GEBCO,N Robinson,NCEAS,NLS,OS,NMA,Geodatastyrelsen and the GIS User Community
Powered by Esri

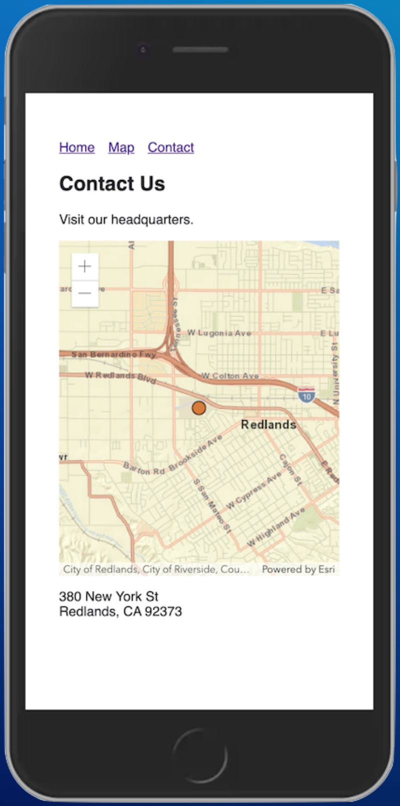
When to use esri-loader?

- Rapid prototyping, hackathons
- Your (hipster) tools have trouble with `@arcgis/core`

Example: esri-gatsby

ArcGIS API + Gatsby

Example: esri-gatsby


Home Map Contact

Contact Us

Visit our headquarters.

380 New York St
Redlands, CA 92373

Map Component

```
<Map latitude={34.0573} longitude={-117.1949} />
```

Conclusion

esri | THE SCIENCE OF WHERE

Please provide your feedback for this session by clicking on the session survey link directly below the video.