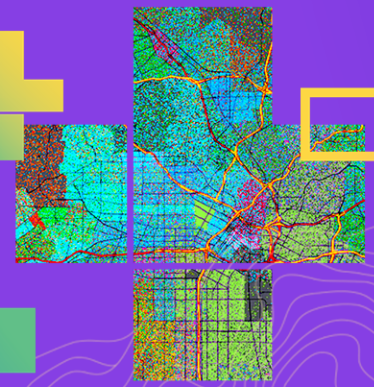
# Node.js and browser applications with ArcGIS REST JS

Allison Davis @araedavis

Noah Mulfinger @noahmulfinger

2020 ESRI DEVELOPER SUMMIT | Palm Springs, CA

slides: bit.ly/3aLDzaz



# Agenda

1. 🌐 - What is ArcGIS REST JS? Why?
2. 👩‍🚀 - Who is using it? For what?
3. 📆 - What's new in 2020?
4. 💯 - Common Patterns
5. 🤹‍♂️ - Demos/code (to learn how it works)

# Open Source on GitHub

Code 🎛️ github.com/Esri/arcgis-rest-js

Doc 📚 esri.github.io/arcgis-rest-js

@esri/arcgis-rest-js helps you talk

to ArcGIS Online and Enterprise

from modern browsers and Node.js.

# Vanilla `XMLHttpRequest`

```
// construct the url yourself and don't forget to tack on f=json
const url = "https://www.arcgis.com/sharing/rest/community/users/dmfenton";

url += "?f=json";

var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
  if (xhr.readyState == XMLHttpRequest.DONE) {
    // make sure JSON response doesnt indicate an error
    if (!xhr.responseText.error) {
      xhr.responseText; // { firstName: "Daniel", description: "open source geodev"
    }
  }
};
xhr.open("GET", url, true);
xhr.send(null);
```

# Vanilla `Fetch`

```
const url = "https://www.arcgis.com/sharing/rest/community/users/dmfenton";

fetch(url, {
  method: "POST", // set the request type
  headers: {
    "Content-Type": "application/x-www-form-urlencoded" // append the right header
  },
  // concat and encode parameters, append f=json
  body: encodeURIComponent("f=json")
})
  .then(response => {
    if (response.ok) {
      return response.json();
    } // dig out the json
  })
  .then(response => {
    // trap for errors inside a 200 response
    if (!response.error) {
      return response;
    }
  });
```

# Even more complexity

- What are all the error codes?
- How do you handle auth?
- How are dates supposed to be encoded?
- Proper encoding for objects?
- How do you manage tokens for federated servers?
- Refreshing authentication when necessary?

# @esri/arcgis-rest-js

```
import { request } from "@esri/arcgis-rest-request";

request(url)
  .then(response) // { firstName: "Daniel", description: "open source geodev" ... }
  .catch((error => {
    if(err.name === "ArcGISAuthError"){
      // handle an auth error
    } else {
      // handle a regular error
    }
  })
```

# Value adds

- appends `f=json` and request headers
- encodes query string parameters
- creates `FormData` (when required)
- clear and informative error handling
- proper parameter encoding
- support for authentication
- ~~display a map~~
- ~~clientside analysis~~

# `request` only expects a url, but it exposes `requestOptions` too.

```
// url, IRequestOptions
request(url, {
  params: {
    // anything you want to pass
    foo: true,
    bar: "baz",
    more: File(),
    num: 999,
    when: Date().now() // etc.
  }
  // httpMethod: "GET",
  // authentication
  // portal,
  // headers,
  // fetch
});
```

# the rest of the API builds on top of `request`

```
import { geocode } from "@esri/arcgis-rest-geocoding";

// assumes you want to use ArcGIS Online
geocode("LAX").then(response); // { ... candidates: [] }

// IRequestOptions is still available
geocode({
  singleLine: "LAX",
  params: {
    forStorage: true
  },
  authentication
});
```

# Goals

- Node.js and (modern) browsers
- a la carte / svelte
- framework agnostic
- shave down the sharp edges
- align with JS ecosystem

# Disclaimer*

- not a product, no roadmap
- work in progress
- scratching our own itch

# Comparison

- *kind of* analogous to ArcGIS API for Python
- **much different** than the ArcGIS API for JavaScript

# In the beginning...

- ArcGIS for Developers
- ArcGIS Hub
- customers!

# As of 2020

- ArcGIS Hub
- ArcGIS for Developers
- Storymaps
- Web AppBuilder (next generation)
- ArcGIS Solutions
- ArcGIS Enterprise
- ArcGIS Analytics for IoT
- Esri UK
- Startups / Partners
- Customers
- You?

# packages 📦!

- `request` 2.8 kB
- `auth` 3.6 kB
- `portal` 5.1 kB
- `feature-layer` 1.3 kB
- `service-admin` 746 B
- `geocoding` 1 kB
- `routing` 642 B

# Common Patterns

when only **one** piece of information is required

```
import { searchItems } from "@esri/arcgis-rest-portal";

searchItems("water").then(response); // { total: 355, etc... }
// or
searchItems({
  query: "water",
  httpMethod: "GET",
  authentication
});
```

you can pass in it in directly.

# if **more** than one piece of information is needed

```
deleteFeatures({
  url: "https://server.arcgis.com/arcgis/rest/
  objectIds: [ 123 ]
})
  .then(response)
```

```
{
  // response
  "deleteResults": [
    {
      "objectId": 123,
      "success": true
    }
  ]
}
```

only an object can be passed in, *extends* `IRequestOptions`

# update who can access an item

```
import { setItemAccess } from "@esri/arcgis-rest-sharing";

setItemAccess({
  id: `fe8`, // which item?
  access: `public`, // who should be able to see it?
  authentication // user allowed to update
}).then(response);
```

ISetItemAccessOptions

Simplified developer experience, even when the underlying logic is complicated

- we ensure the response is *deterministic*
- we figure out *which* url to call (based on role)

# Authentication

```
import { UserSession } from "@esri/arcgis-rest-auth";

// ArcGIS Online credentials
const authentication = new UserSession({ username, password });

// ArcGIS Enterprise credentials
const enterpriseAuth = new UserSession({
  username,
  password,
  portal: `https://gis.city.gov/sharing/rest`
});
```

# `UserSession` keeps track of token expiration

```
const url = `http://geocode.arcgis.com/arcgis/rest/services/World/GeocodeServer/`;

const authentication = new UserSession({ username, password });

request(url, { authentication }).then(response => {
  // the same token will be reused for the second request
  request(url, { authentication });
});
```

and whether or not a server is trusted (federated)

# since DevSummit 2019...

🎉 rest-js v2.0.0! 🎉

(plus 20 additional releases 🚀)

# What's new in v2+

- SearchQueryBuilder
- Improved paging
- `setDefaultRequestOptions()` and `withOptions()`
- Package and type reorganization

# Building search queries

```
// 1.x
const q =
  "Trees AND owner: US Forest Service AND (type: 'Web Mapping Application' OR type:

// 2.x
const q = new SearchQueryBuilder()
  .match("Trees")
  .and()
  .match("US Forest Service")
  .in("owner")
  .and()
  .startGroup()
  .match("Web Mapping Application")
  .in("type")
  .or()
  .match("Mobile Application")
  .in("type")
  .endGroup();
```

# One portal package to rule them all

```
// 1.x
npm install @esri/arcgis-rest-items &&
@esri/arcgis-rest-users &&
@esri/arcgis-rest-groups &&
@esri/arcgis-rest-sharing

// 2.x
npm install @esri/arcgis-rest-portal
```

# Packages install types automatically

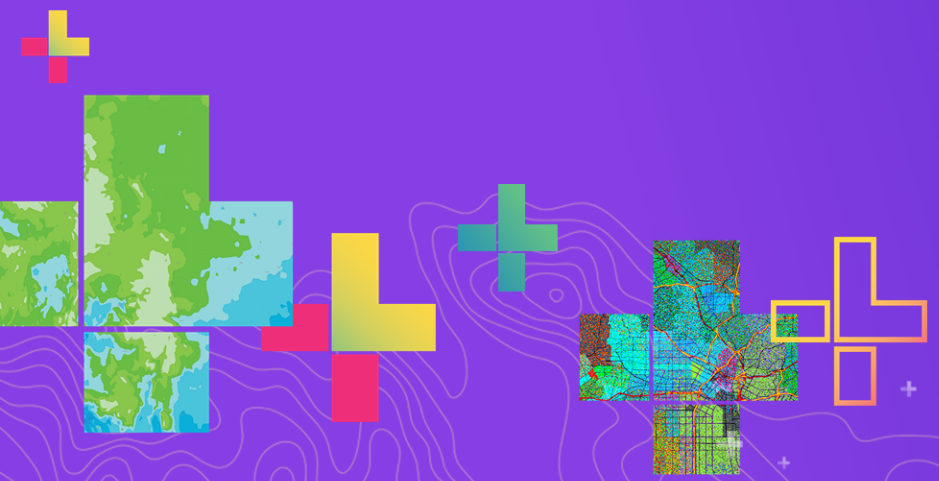
```
// 1.x
import { IPoint } from "@esri/arcgis-rest-common-types";
import { reverseGeocode } from "@esri/arcgis-rest-geocoder";

reverseGeocode({ x: 34, y: -118 } as IPoint);

// 2.x
import { IPoint, reverseGeocode } from "@esri/arcgis-rest-geocoding";

reverseGeocode({ x: 34, y: -118 } as IPoint);
```

Check out the release notes for the full list

# Demo

## OAuth in Browser

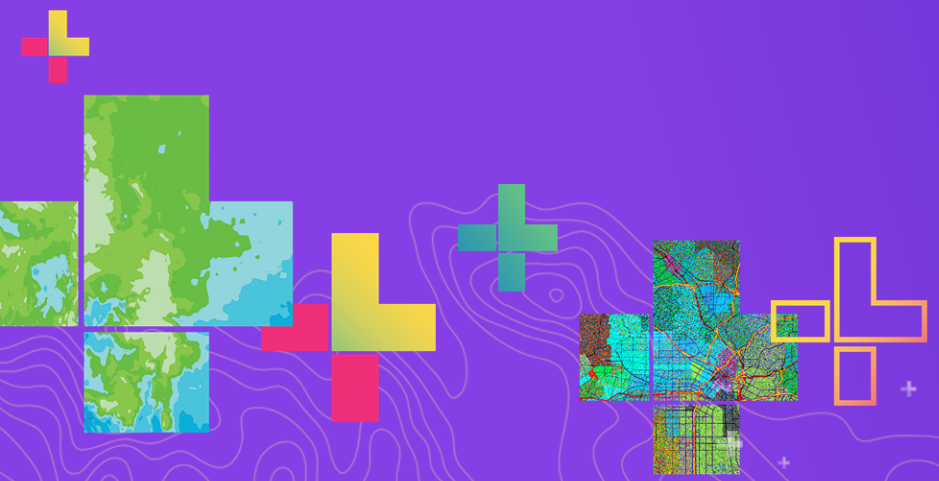
- Auth package
- rest-js via CDN



# Demo

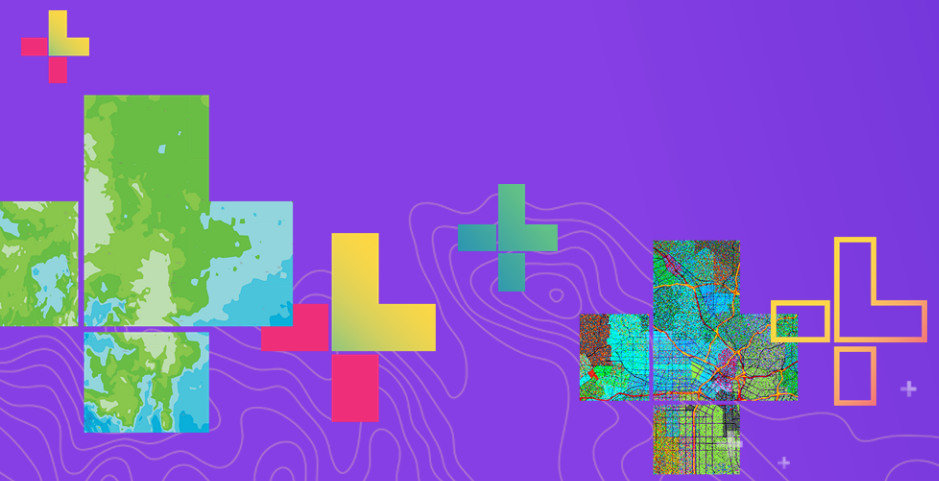
## React Component

- Geocoding package
- Downshift



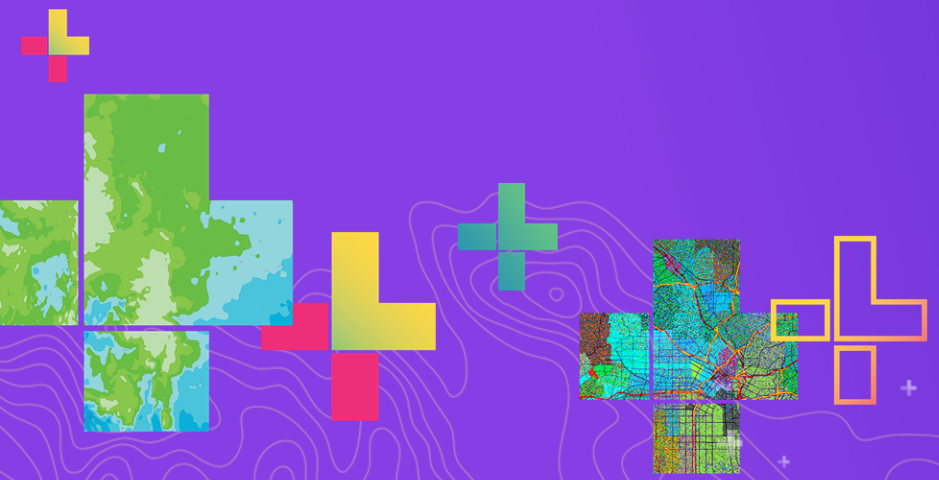
# Demo

## JS API Integration with Angular



# Demo

## Node.js



# Resources 📚

- Link to slides
- GitHub repo
- Docs site
- Demo at Observables

# More Demos 💻

- Sapper
- Web Components with Stencil
- Lamda Functions

esri

THE SCIENCE OF WHERE