

API'S



OUTLINE

- Course review
- What are API's and what do they do?
- Making an API Call
 1. Build your URL
 2. Encode the URL
 3. Process the content
 - Spatial Data with Esri
- Geocode Example
- Shiny Example



Survey Monkey



Whats an API,
and what does it do?

API EXAMPLES

- WPRDC
- Census
- Geocoders
- Esri Online Datasets
- Online Weather APIs
- Sport Score API
- And more!

API'S

- Stands for: Application Programming Interface
- There are many kinds of API's
 - *Web service*
 - SOAP, XML-RPC, JSON-RPC, and **REST**
 - *WebSocket*
 - *Library-based*
 - *Class-based*
 - *OS functions and routines*
 - *Object remoting*
 - *Hardware*

REST API'S

- End points - different URL's that tell the webserver what data you would like
- It's essentially a website where you request different "end points"
- There are 5 types of Requests you can make
 - GET (what we will use the most in this course)
 - POST (*sometimes necessary for authentication, if you're trying to write data somewhere*)
 - PUT
 - PATCH
 - DELETE

Making an API Call

THE STEPS

1. Build your URL
2. Encode the URL
3. Process the content
4. Transform to a usable format

1. BUILDING YOUR QUERY

Many tools that make life easier:

- *Insomnia*
- *Advanced REST Client*
- *PostMan*
- *And others...*

GET ▾

https://data.wprdc.org/api/3/action/datastore_search_sql

Send

Body ▾

Auth ▾

Query ¹

Header

Docs

URL PREVIEW

https://data.wprdc.org/api/3/action/datastore_search_sql?s
ql=SELECT%20DISTINCT(%22type%22)%20from%20%22fbb50b02-2879-4
7cd-abea-ae697ec05170%22

📄

≡ sql

✎ 73 bytes

▾

✓

🗑

⚙

New name

New value

WPRDC API Call in Insomnia

THE STEPS

1. Build your URL
2. Encode the URL
3. Process the content
4. Transform to a usable format



DEMO

```
URLencode("someString", repeated = TRUE)
```

THE STEPS

1. Build your URL
2. Encode the URL
3. Process the content
4. Transform to a usable format

CONTENT

- ▶ Any API call will have multiple portions of it.
- ▶ 2 most important are:
 - ▶ Content
 - ▶ status_code

GETTING TO THE CONTENT

- ▶ Most API calls you will be making are GET requests.

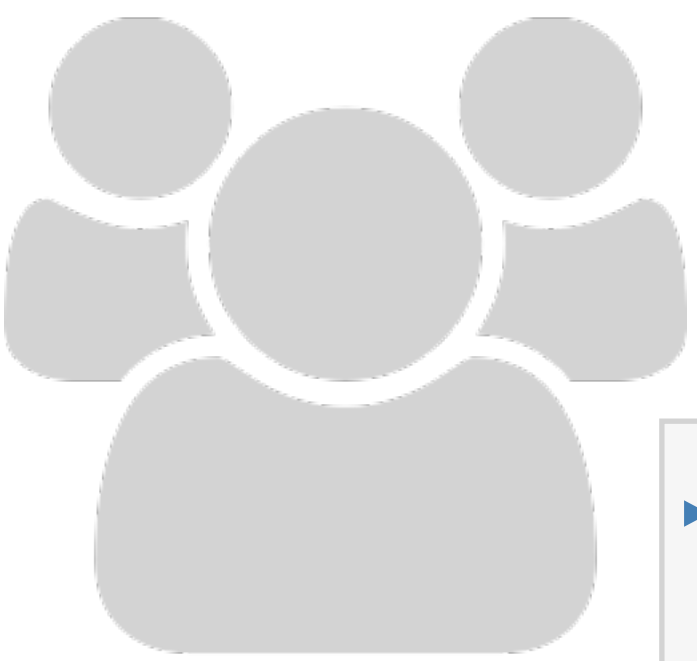
```
get <- httr::GET("encodedURL")  
c <- jsonlite::fromJSON(content(get, "text"))
```

- ▶ Arguments you may need:
 - ▶ \$something after fromJSON function
 - ▶ flatten=TRUE

ERRORS

- ▶ Status codes indicate the result of the HTTP request.
 - ▶ **100's** - *info*
 - ▶ **200's** - *success*
 - ▶ **300's** - *redirection*
 - ▶ **400's** - *client error (you messed up)*
 - ▶ **500's** - *server error (something went wrong on their end, but you still could have messed up)*

EXERCISE



- ▶ Open `exercises/api_practice.Rmd` and use the chunk labeled “Blotter”
 - ▶ Like last class generate an API call that downloads all of the data from the City of Pittsburgh Police Blotter
 - ▶ It might be easier to build the query in Insomnia or something else first
 - ▶ Stretch: After you have built a query that calls all of the data, add a group by or filter of some kind

10_m 00_s

Spatial Data

Query: propertyowner6 (ID: 0)

Where:	<input type="text"/>
Object IDs:	<input type="text"/>
Time:	<input type="text"/>
Input Geometry:	<input type="text"/>
Geometry Type:	Envelope
Input Spatial Reference:	<input type="text"/>
Spatial Relationship:	Intersects
Result Type:	None
Distance:	0.0
Units:	Meters
Return Geodetic:	<input type="radio"/> True <input checked="" type="radio"/> False
Out Fields:	<input type="text"/>
Return Geometry:	<input checked="" type="radio"/> True <input type="radio"/> False
Return Centroid:	<input type="radio"/> True <input checked="" type="radio"/> False
Feature Encoding:	esriDefault
Geometry MultiPatch Option:	xyFootprint
Max Allowable Offset:	<input type="text"/>
Geometry Precision:	<input type="text"/>
Output Spatial Reference:	<input type="text"/>
Datum Transformation:	<input type="text"/>
Apply VCS Projection:	<input type="radio"/> True <input checked="" type="radio"/> False
Return IDs Only:	<input type="radio"/> True <input checked="" type="radio"/> False
Return Unique IDs Only:	<input type="radio"/> True <input checked="" type="radio"/> False
Return Count Only:	<input type="radio"/> True <input checked="" type="radio"/> False

SQL like where statement to get one the data you want

Same as the select portion of a SQL query

GETTING SPATIAL DATA

- For ESRI API's so long as your format is set to GEOJSON...

```
data <- read0GR("encodedURL")
```

- Its that easy

EXERCISE



- ▶ Open `exercises/api_practice.Rmd` and go to the chunk labeled “Esri”
 - ▶ Look at the fields on the May 2019 Election layer from the Allegheny County Esri API: https://services1.arcgis.com/vdNDkVykv9vEWFx4/ArcGIS/rest/services/Allegheny_County_Polling_Places_May2019/FeatureServer/0
 - ▶ Get all of the polling places in just the City of Pittsburgh and load it into R from the URL

10_m 00_s



SOLUTION

Solutions to both of today's exercises are in:
`api_practice_solutions.R`

THE STEPS

1. Build your URL
2. Encode the URL
3. Process the content
4. Transform to a usable format

Geocode

Example



DEMO

alco_geocode.R

Shiny

Example



DEMO

app/311_dashboard.R

API'S





**That's how extra credit
is supposed to feel.**