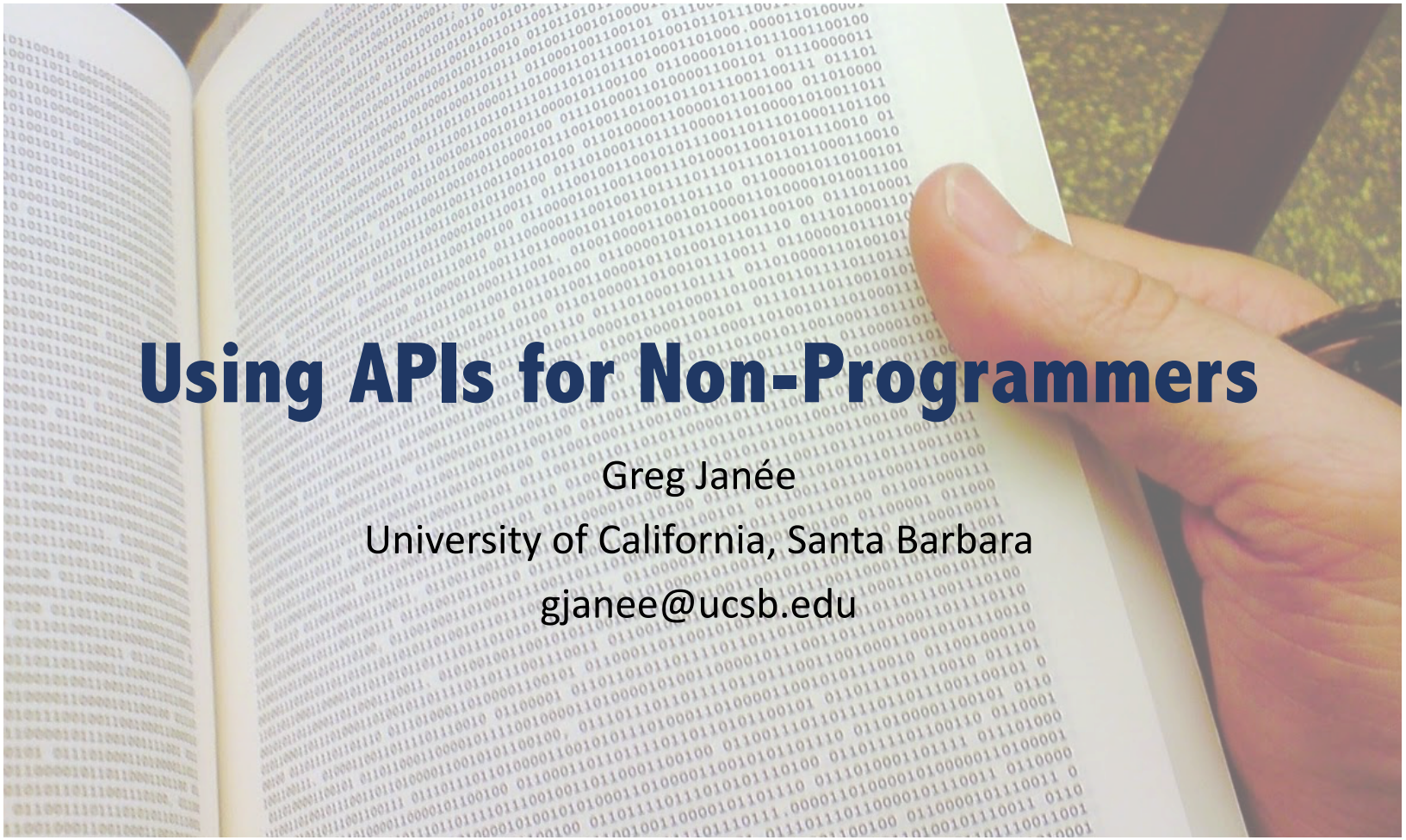

Using APIs for Non-Programmers

Greg Janée
University of California, Santa Barbara
gjanee@ucsb.edu

These slides at: **https://tinyurl.com/api-howto**

Why?

- Lots of APIs available!
  - Repository systems, publication/preprint systems, museum catalogs, ...
  - https://www.programmableweb.com
- Get information not otherwise available
- Get information in compute-friendly form
- Prelude to automation
- *It's now possible!*

Outline

- Terminology
- Basic process
- Three methods
  - web browser
  - curl
  - swishy desktop application
- How to interpret API documentation
- Handling paging of results
- Further steps

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

- HTTP verbs
  - GET – 99% of time
  - POST – form submission

Terminology

- Parts of a URL
  - protocol://server/path?query#fragment
  - https://www.google.com/search?q=aardvark#result2

- HTTP verbs
  - GET – 99% of time
  - POST – form submission

- Query formatting
  - param1=value1&param2=value2&...
  - q=drosophila&type=dataset

Basic process

1. Construct a URL
   - Basic access point + request

2. Get back something readable
   - JSON nowadays
   - But maybe XML, CSV, other text format

3. View JSON

“REST” or “RESTful” API

Example: determining a DOI's registrar

- DOI identifiers uniquely identify publications, datasets, ...
  - 10.3886/ICPSR36422
  - doi:10.3886/ICPSR36422
  - https://doi.org/10.3886/ICPSR36422
- Multiple registration agencies
  - Crossref, DataCite, mEDRA, ...
- Crossref API
  - GET https://api.crossref.org/works/[doi]/agency
  - GET https://api.crossref.org/works/10.3886/ICPSR36422/agency

Method 1: web browser

- Visit URL

- Copy/paste JSON into JSON viewer
  - search for “JSON viewer”

Method 2: curl

- Standard command line tool
- Lots of options

```
> curl 'http://api.crossref.org/works/10.3886/ICPSR36422/agency'
{"status":"ok","message-type":"work-agency","message-
version":"1.0.0","message":{"DOI":"10.3886\/icpsr36422","agency":{"
id":"datacite","label":"DataCite"}}}
>
```

Method 3: Postman

- Free desktop application
- https://www.postman.com

Example 2: art museum

- Metropolitan Museum of Art API
  - Documentation at https://metmuseum.github.io
  - API endpoint: https://collectionapi.metmuseum.org
- Task: find highlight vases
- Notice how Postman formats query parameters
  - Manually, use https://ascii.cl/url-encoding.htm
- If POST request is required
  - Don’t use URL query parameters
  - Use body
  - Use form-data or x-www-form-urlencoded option
  - Enter query parameters

Example 3: preprint archive

- EarthArXiv preprint, hosted by OSF
  - Documentation at https://developer.osf.io
  - API endpoint: https://api.osf.io/v2

- Task: find papers on geomorphology

- Note paging links at bottom of JSON

Further study: automation

- Needed to, for example, obtain information on all Met vases

- Library Carpentry lesson on web scraping
  - https://librarycarpentry.org/lc-webscraping/
  - N.B.: automating API calls = web scraping, but vastly easier