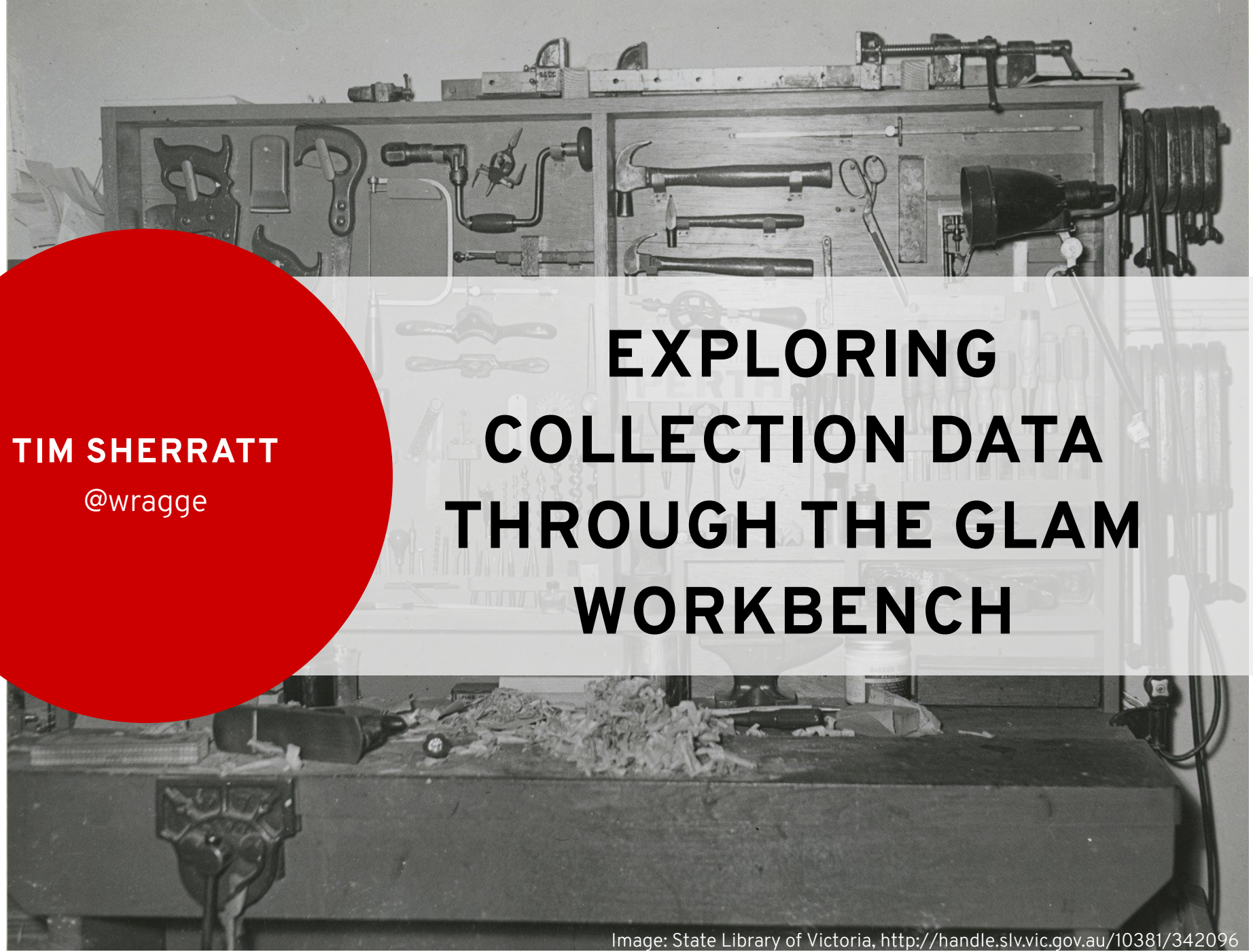
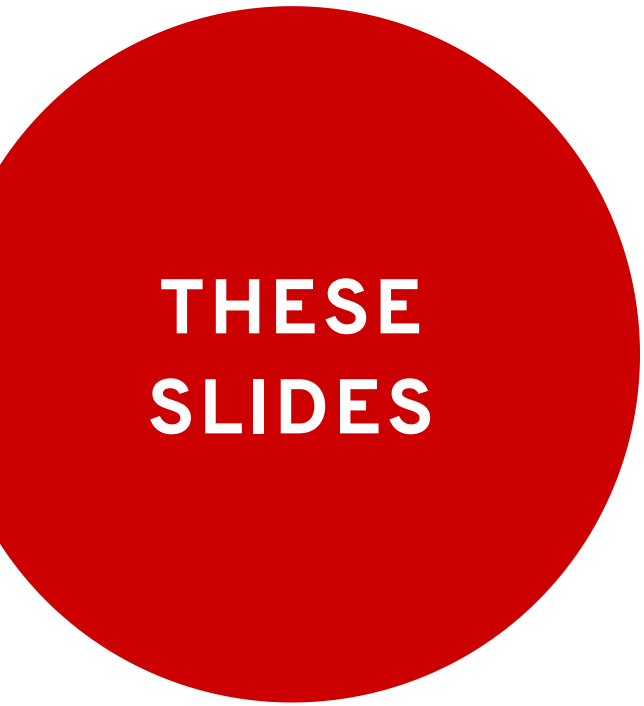


EXPLORING COLLECTION DATA THROUGH THE GLAM WORKBENCH

TIM SHERRATT

@wragge

Image: State Library of Victoria, http://handle.slv.vic.gov.au/10381/342096



https://slides.com/wragge/2021-explore-collection-data/

Rough notes of my talk...

COLLECTIONS AS DATA?

See also: https://collectionsasdata.github.io/

SEARCH IS FAMILIAR

TROVE

ABOUT HELP NEWS PARTNERS SIGN UP LOGIN

Explore Categories Community Research First Australians

Home / Search results

All | Newspapers & Gazettes | Magazines & Newsletters | Images, Maps & Artefacts | Research & Reports | Books & Libraries | Diaries, Letters & Archives | Music, Audio & Video | People & Organisations | Websites | Lists

Newspapers & Gazettes

radio

Advanced search

NEWSPAPERS & GAZETTES

3,430,613 total results Sort by: Relevance

RADIO.

Article - The Longreach Leader (Qld. : 1923 - 1954) Friday 26 September 1924 - Page 14

... prophetic imagination. Solleys Ltd. advertise in this issue that they are licensed dealers in all **radio** ... **RADIO** Are you interested in wireless? Have you considered what is in the,air for you, and at your ... 111 words

Text corrected by 1 Voluntrove

RADIO.

Article - The West Australian (Perth, WA : 1879 - 1954) Tuesday 15 September 1925 - Page 6

... **RADIO**. Mr. H. Brooghton Jensen, who was recently engaged to make an examina-tion of the **Radio** mime ... 68 words

Text corrected by 1 Voluntrove

REFINE YOUR RESULTS

Type

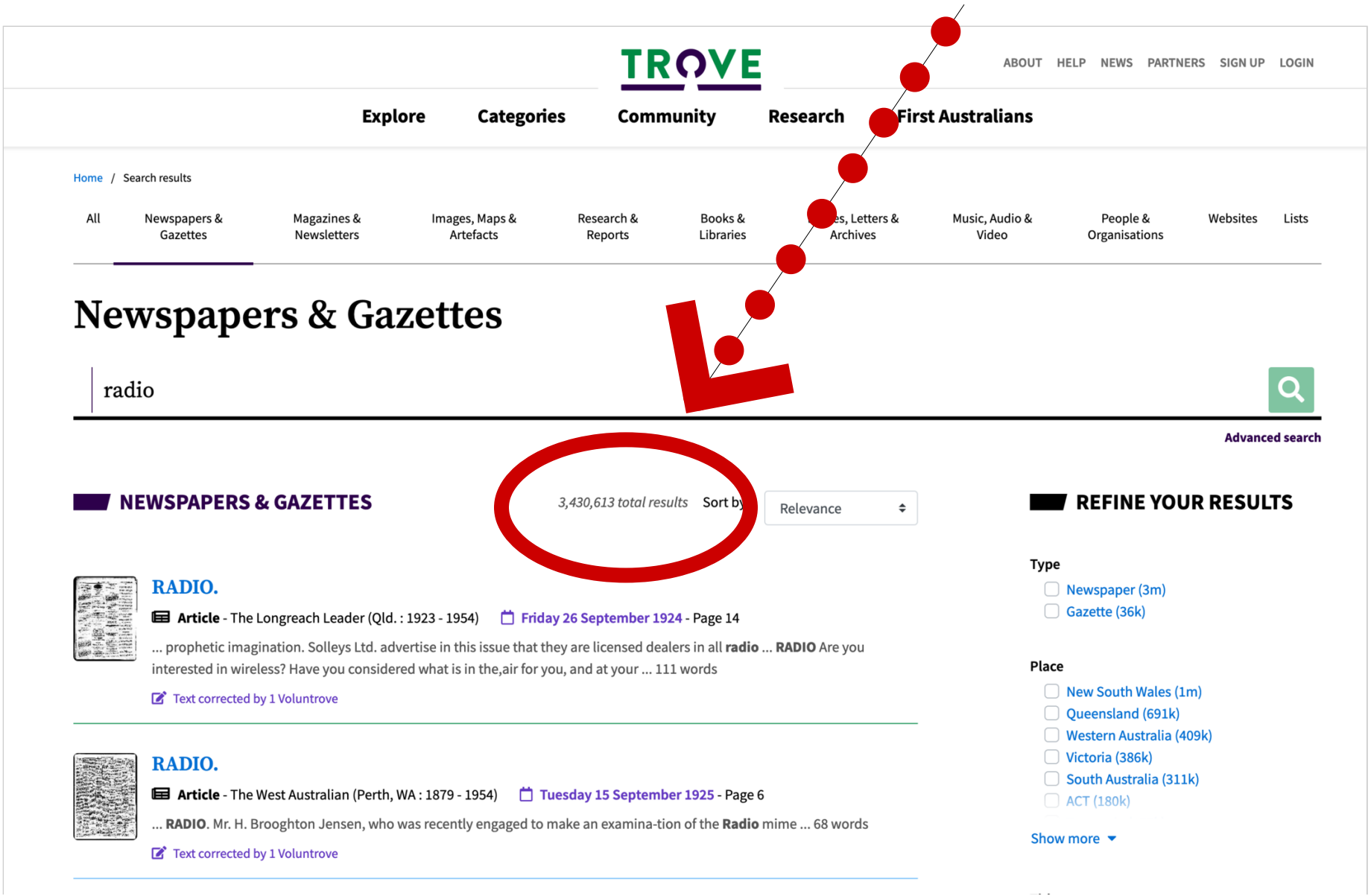
- Newspaper (3m)
- Gazette (36k)

Place

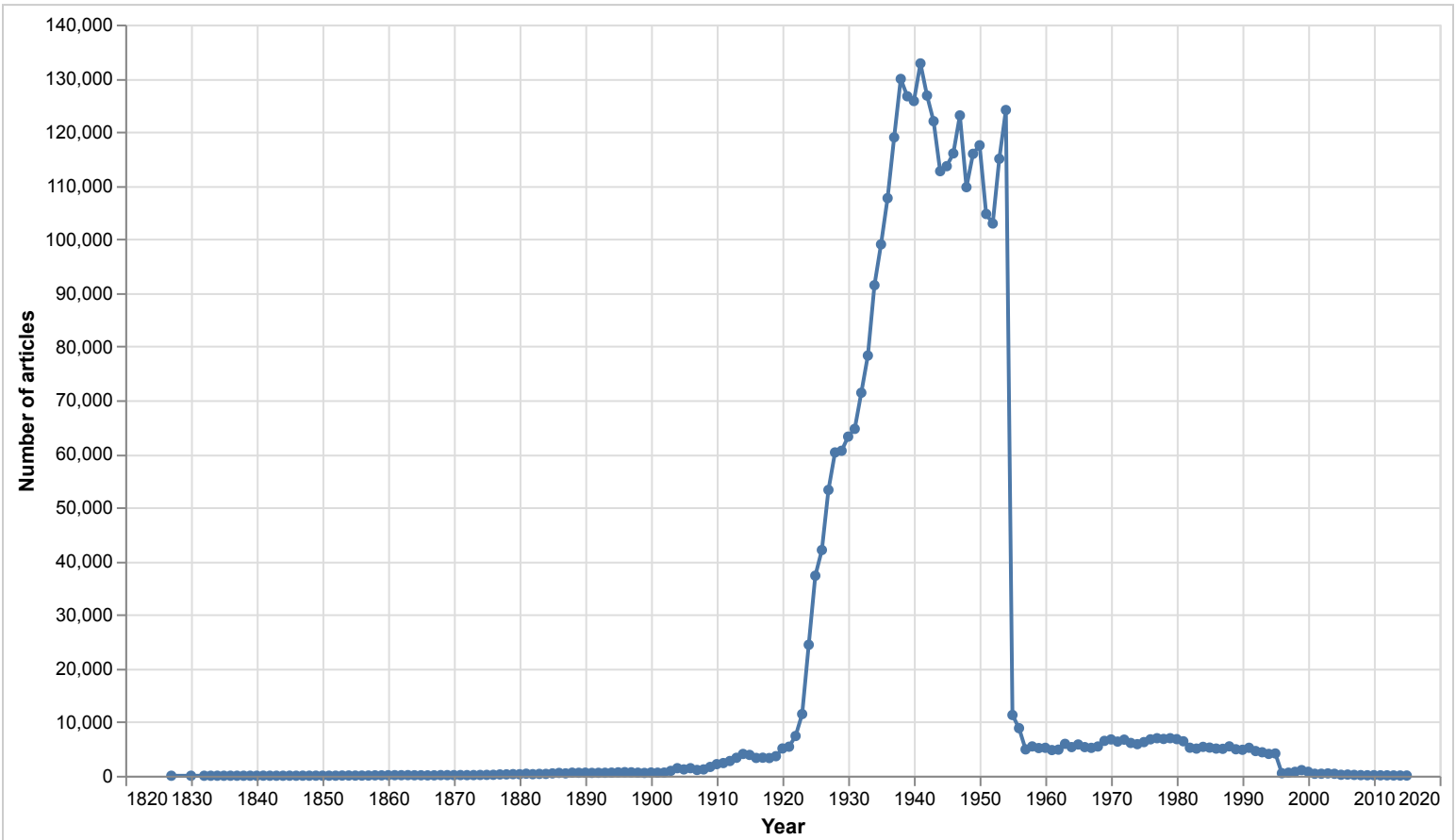
- New South Wales (1m)
- Queensland (691k)
- Western Australia (409k)
- Victoria (386k)
- South Australia (311k)
- ACT (180k)

Show more

BUT WHAT ABOUT THE OTHER 3M?

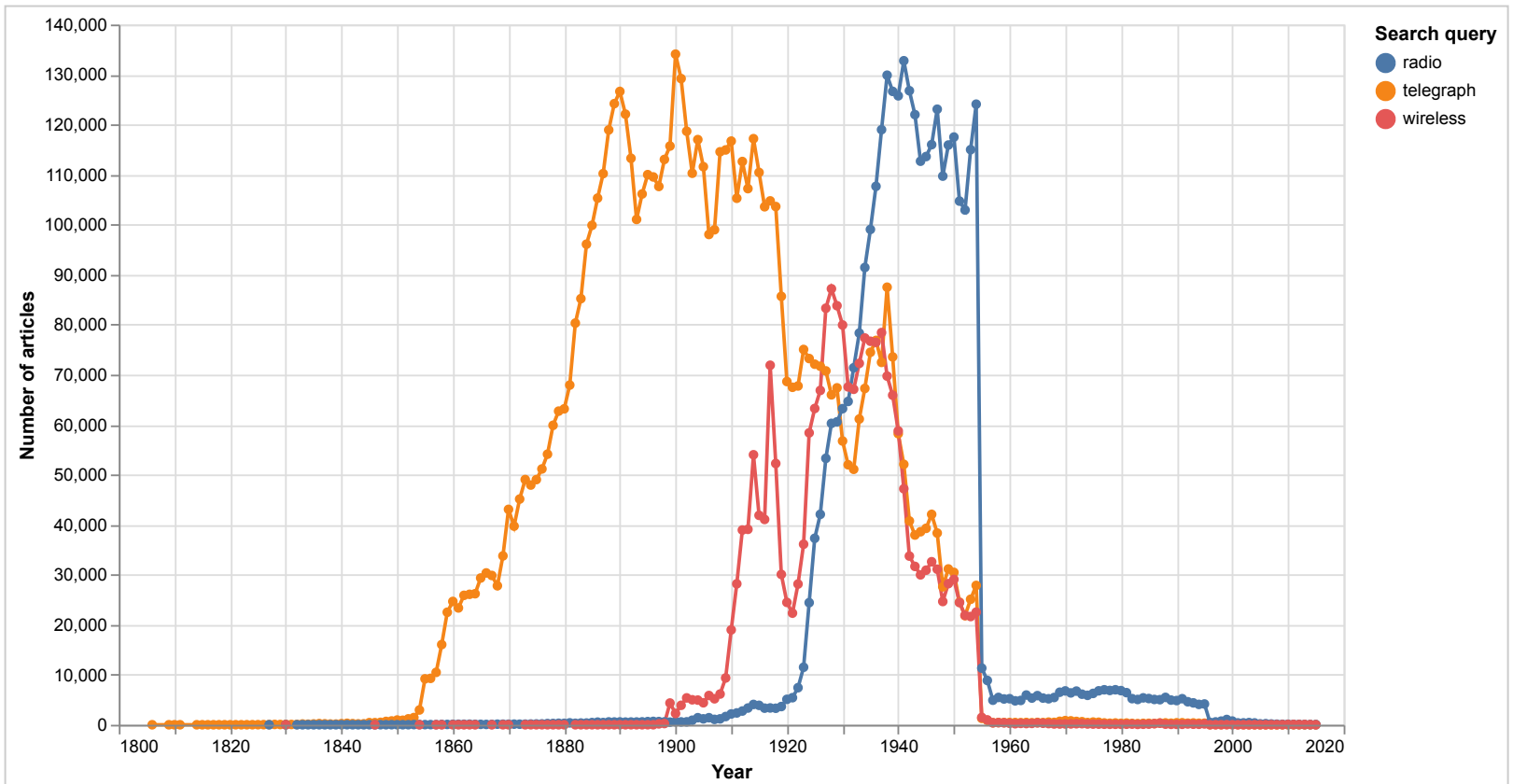


SAME SEARCH



DIFFERENT VIEW

MORE SEARCHES

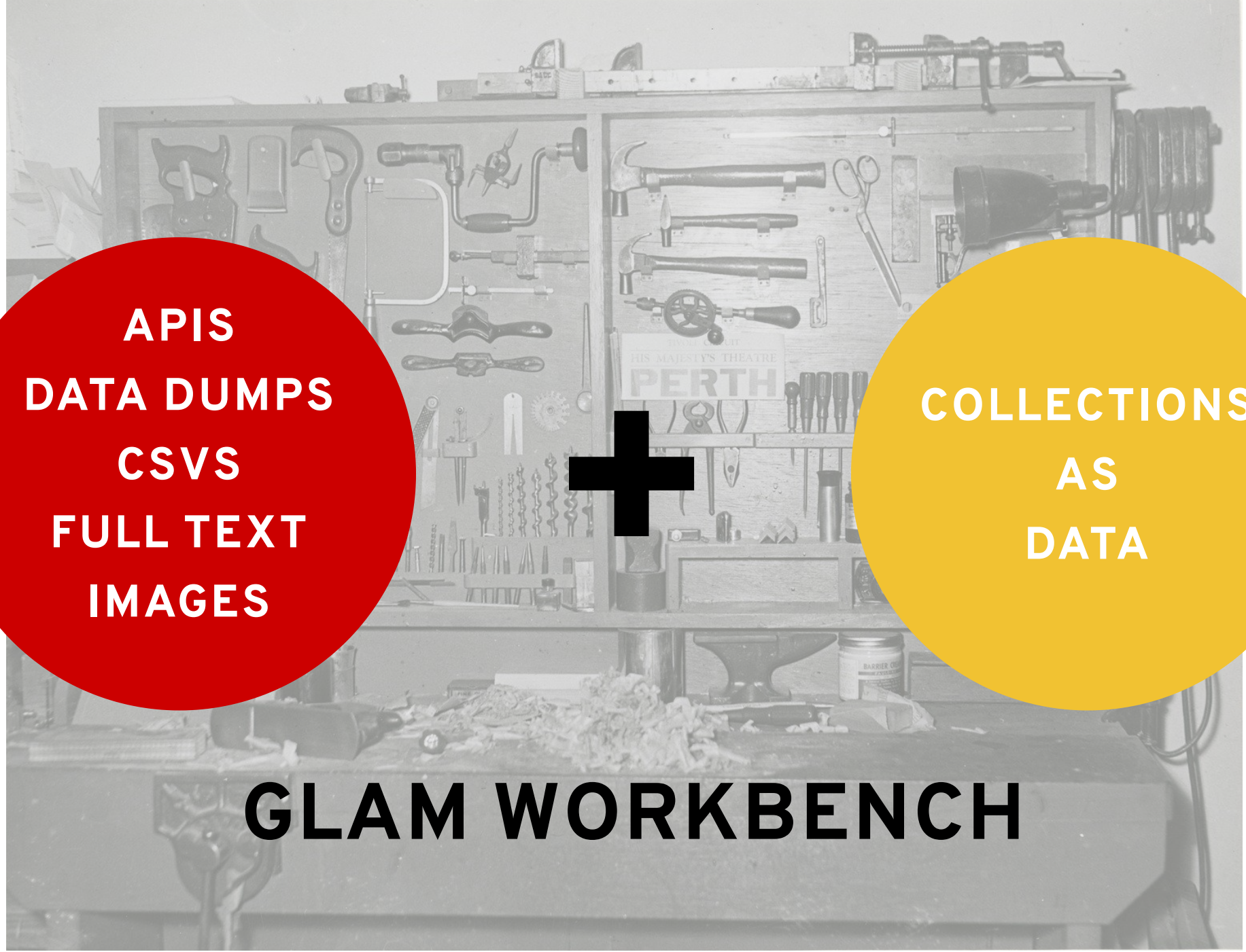


DIFFERENT QUESTIONS!


APIS
DATA DUMPS
CSVS
FULL TEXT
IMAGES

?


COLLECTIONS
AS
DATA


APIS
DATA DUMPS
CSVS
FULL TEXT
IMAGES
+
COLLECTIONS
AS
DATA
GLAM WORKBENCH

GLAM WORKBENCH

GLAM Workbench | Search | GLAM-Workbench 41 Repositories

GLAM Workbench

- Home
- About
- Help
- Data sources
- Trove
- DigitalNZ
- Archives
- Libraries
- Web Archives
- Museums
- Government
- Suggest a topic
- Ask a question

chat on gitter

Welcome to the GLAM Workbench

Here you'll find a collection of tools, tutorials, examples, and hacks to help you work with data from galleries, libraries, archives, and museums (the GLAM sector). The primary focus is Australia and New Zealand, but new collections are being added all the time. Let me know if there's some GLAM data you'd like me to explore – suggestions are always welcome!

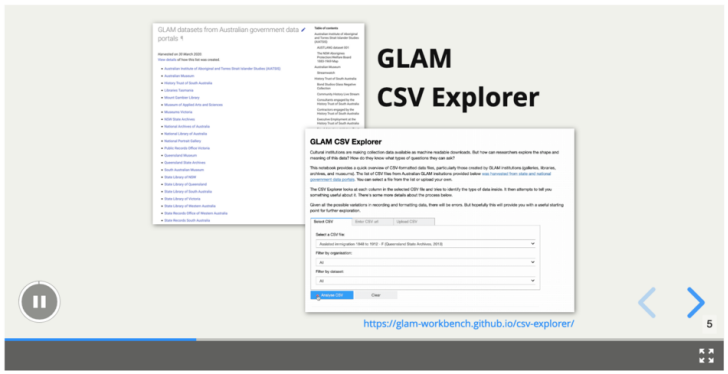


Table of contents

Quick start

The resources in the GLAM Workbench are created and shared as Jupyter notebooks. Jupyter lets you combine narrative text and live code in an environment that encourages you to learn and explore. Jupyter notebooks run in your browser, and you can get started without installing any software!

If you want to dive straight in, just have a look around the site and click on one of the links that says '**Run live on Binder**'. This will open the notebook, ready to use, in a customised computing environment using the Binder service.

If that seems too scary, here's some first steps to get you started.

https://glam-workbench.net/

IS

- tools, tutorials, examples, hacks
- live code
- editable, reusable, hackable
- openly licensed

IS NOT

- coding 101
- finished or perfect

NOT JUST HOW, BUT WHY...

- **possibilities** – why should I be interested?
- **starting points** – can you give me an example I can use?
- **pathways** – where do I go next?

JUPYTER NOTEBOOKS

CODE

Let's display the results as a simple line chart using Altair

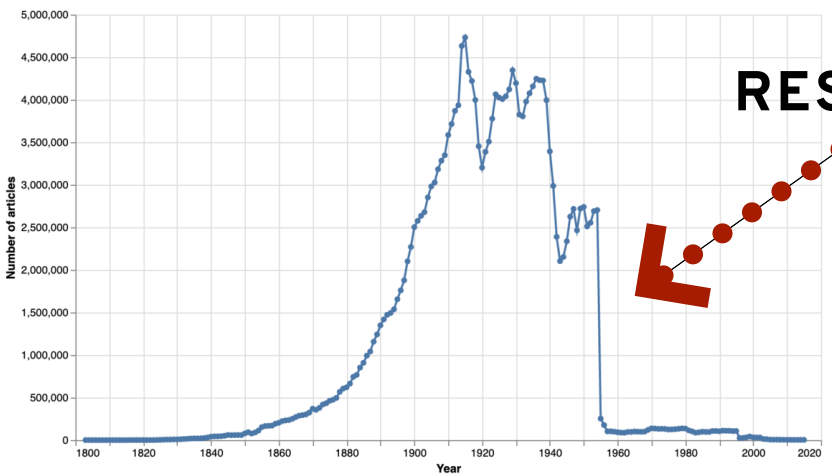
In [11]:
```
# Feed Altair our dataframe
alt.Chart(df_total).mark_line(point=True).encode(

    # Years along the X axis
    x=alt.X('term:Q', axis=alt.Axis(format='c', title='Year')),

    # Number of articles on the Y axis (formatted with thousands separators)
    y=alt.Y('total_results:Q', axis=alt.Axis(format=',d', title='Number of articles')),

    # Use tooltips to display the year and number of articles when you hover over a point
    tooltip=[alt.Tooltip('term:Q', title='Year'), alt.Tooltip('total_results:Q', title='Articles', format=',')]
).properties(width=700, height=400)
```

Out[11]:



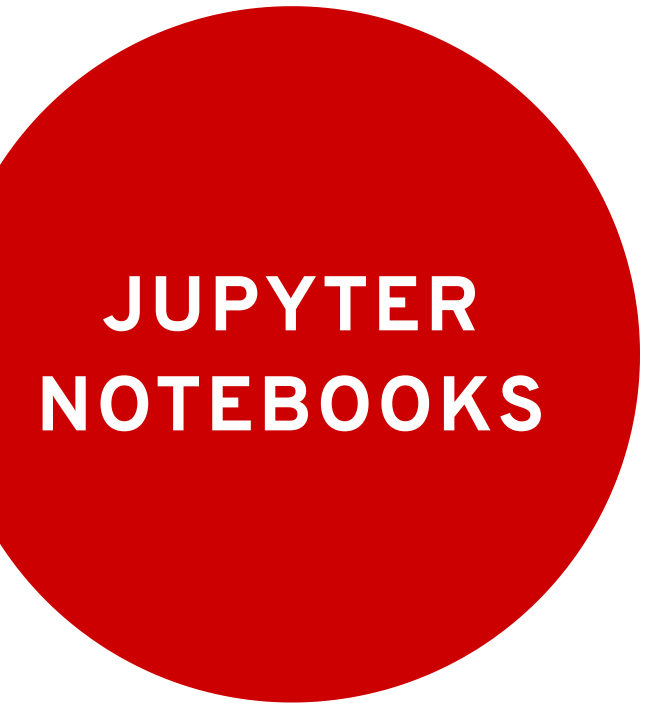
RESULTS

Hmmm, that *is* interesting. There's a significant peak in the number of articles around 1915. Why might that be? Were there more newspapers? Were more articles written because of the war?

Nope. It's because of funding and digitisation priorities. Not all Australian newspapers are in Trove. Some have been lost, and many are just waiting to be digitised. Funding is always limited, so priorities have to be set. In the lead up to the centenary of World War I, it was decided to focus on digitising newspapers from that period. This chart reflects those priorities. This is not the number of newspaper articles published in Australia, it's the number of newspaper articles that have been digitised and made available through Trove. It's important to remain aware of this as you use Trove.

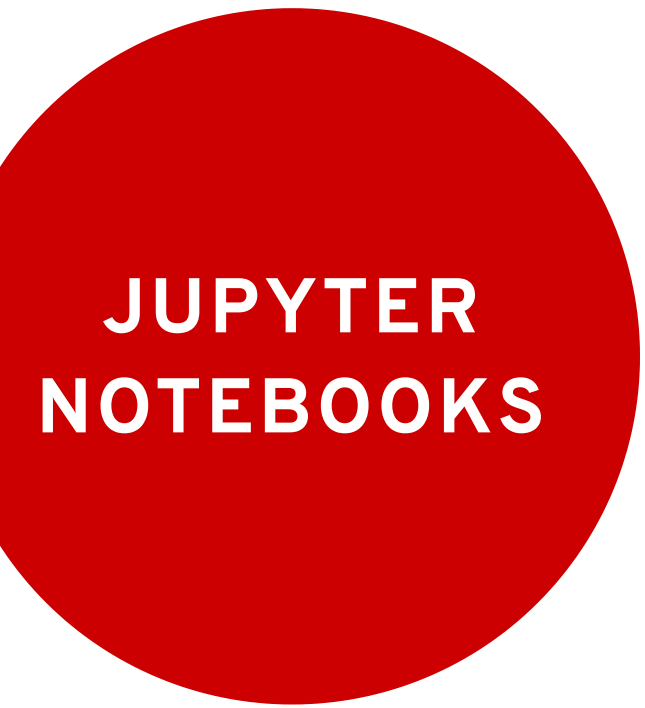
DISCUSSION

WHY JUPYTER?



- Computing in your browser
- A **computational narrative** – combine text, images, code & more
- A standard format – use on different platforms
- See Introduction to Jupyter Notebooks

GLAM EXAMPLES



- Biblioteca Virtual Miguel de Cervantes Labs
- British Library
- National Library of Scotland
- Library of Congress
- More...

JUPYTER CAN BE...

NOT JUST NOTEBOOKS

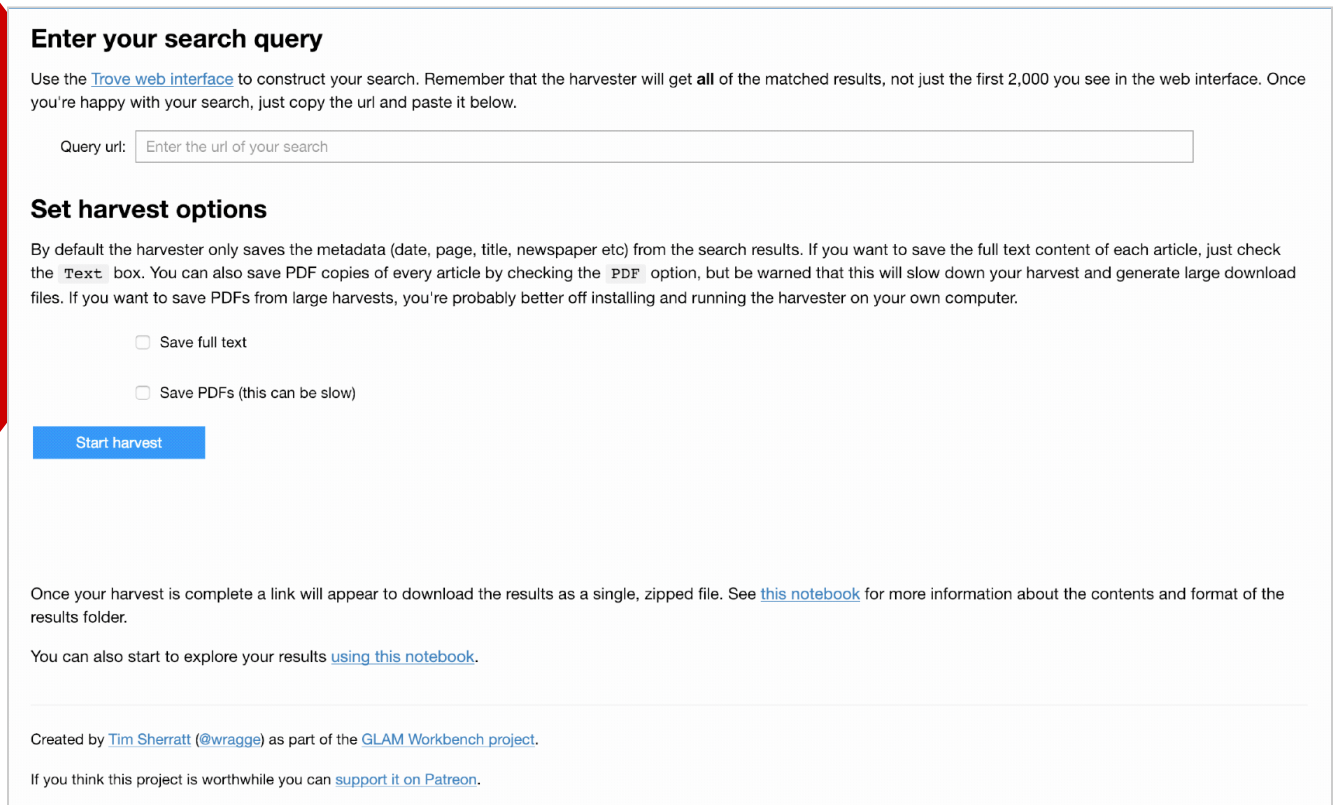
- used on multiple platforms
- presented in different formats
- changed by extensions
- tailored to different users

FINDING GLAM DATA

PACKAGING DATA

(create collections of newspaper articles)

TROVE NEWSPAPER HARVESTER



Enter your search query

Use the Trove web interface to construct your search. Remember that the harvester will get **all** of the matched results, not just the first 2,000 you see in the web interface. Once you're happy with your search, just copy the url and paste it below.

Query url: Enter the url of your search

Set harvest options

By default the harvester only saves the metadata (date, page, title, newspaper etc) from the search results. If you want to save the full text content of each article, just check the `Text` box. You can also save PDF copies of every article by checking the `PDF` option, but be warned that this will slow down your harvest and generate large download files. If you want to save PDFs from large harvests, you're probably better off installing and running the harvester on your own computer.

- [ ] Save full text
- [ ] Save PDFs (this can be slow)

Start harvest

Once your harvest is complete a link will appear to download the results as a single, zipped file. See this notebook for more information about the contents and format of the results folder.

You can also start to explore your results using this notebook.

Created by Tim Sherratt (@wragge) as part of the GLAM Workbench project.

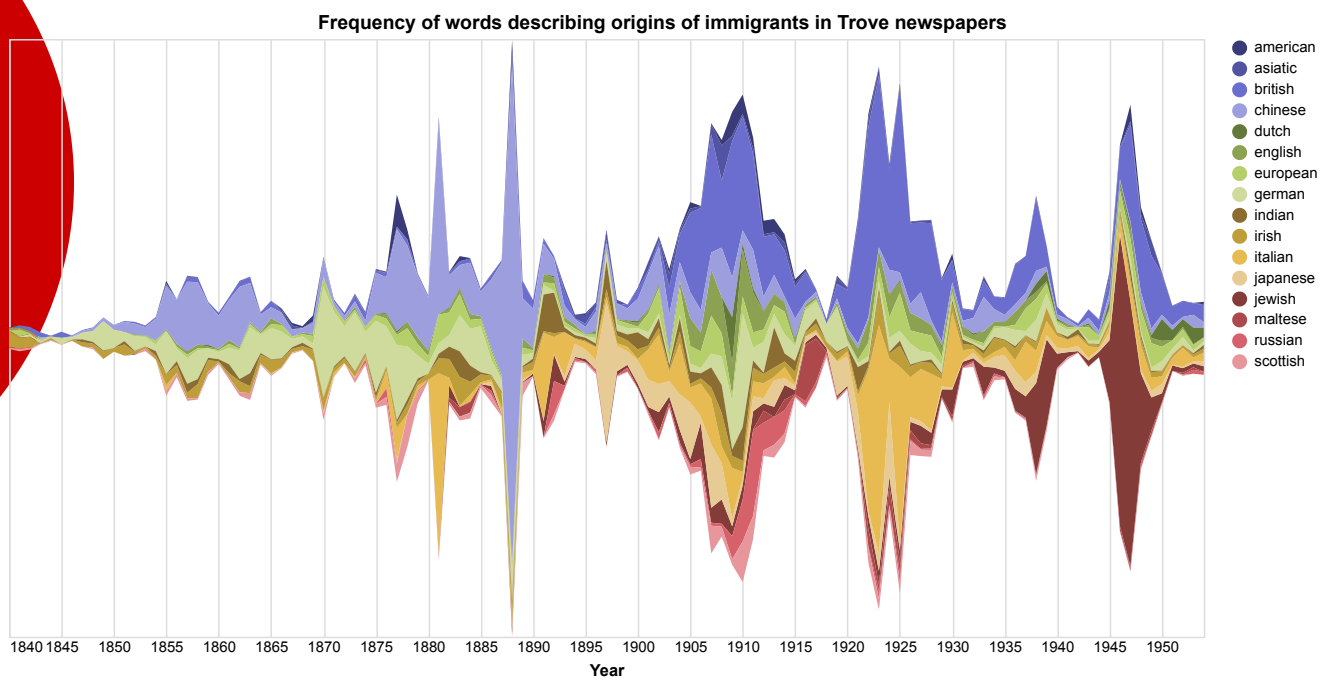
If you think this project is worthwhile you can support it on Patreon.

https://glam-workbench.net/trove-harvester/

PACKAGING DATA

(create collections of newspaper articles)

TROVE NEWSPAPER HARVESTER



http://timsherratt.org/blog/who-belongs/

EXTRACT METADATA

(no API? you can always try screen-scraping!)

NATIONAL ARCHIVES OF AUSTRALIA

5. Harvesting a complete set of (less than 20,000) results

Ok, we've learnt how to create a search and get back some data, but only getting the first 20 results is not so useful. What if our search contains hundreds or thousands of items? How do we get them all?

To save everything, we have to loop through each page in the result set, saving the results as we go. The functions below do just that.

But wait! You might have noticed that RecordSearch only displays results for searches that return fewer than 20,000 items. Because the screen scraper is just extracting details from the RecordSearch web pages, the 20,000 limit applies here as well. If your search has more than 20,000 results, you'll need to narrow it down using additional parameters.

The main function below is `harvest_items()`. You just give it any of the search parameters listed above. It will loop through all the pages in the result set, saving the items to a simple JSON database using TinyDB.

The database will be created in the `data` directory. It's name will include a timestamp, identifying the time at which the harvest was started. For example `db-items-1567492794.json`. There are more functions for using and managing the db files below.

In [303]:
```
def get_total_results(client, **kwargs):
    '''
    Get the total number of results returned by a search.
    '''
    try:
        # Get the first page of results, passing digitised=Flase to speed things up
        results = client.search(digitised=False, **kwargs)

        # Get the total number of results
        total = results['total_results']

    # Uh oh there are more than 20,000 results
    except TooManyError:
        print('There are more than 20,000 results.')
        total = None
    return total

def harvest_items(start=1, db_path=None, check_duplicates=False, **kwargs):
    '''
    Harvest items from a search and save them to a database.
    Supply any of the search parameters listed above.

    Set check_duplicates to True if you want to check for possible duplicates (probably not necessary in most cases).
    '''
    # Initiate the client
    client = RSSearchClient()

    # Get the total number of results returned by this search
    total_results = get_total_results(client, **kwargs)
```

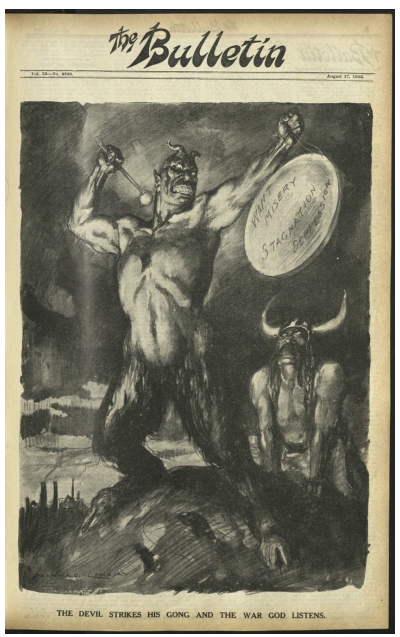
https://glam-workbench.net/recordsearch/


Shhh... it's a SECRET!
These redactions have been recycled from ASIO surveillance files in the National Archives of Australia (NAA: A6119).
NATIONAL ARCHIVES OF AUSTRALIA
https://updates.timsherratt.org/2021/04/21/secrets-and-lives.html

DOWNLOAD IMAGES

(a bit of API, a bit of screen-scraping)

TROVE JOURNALS



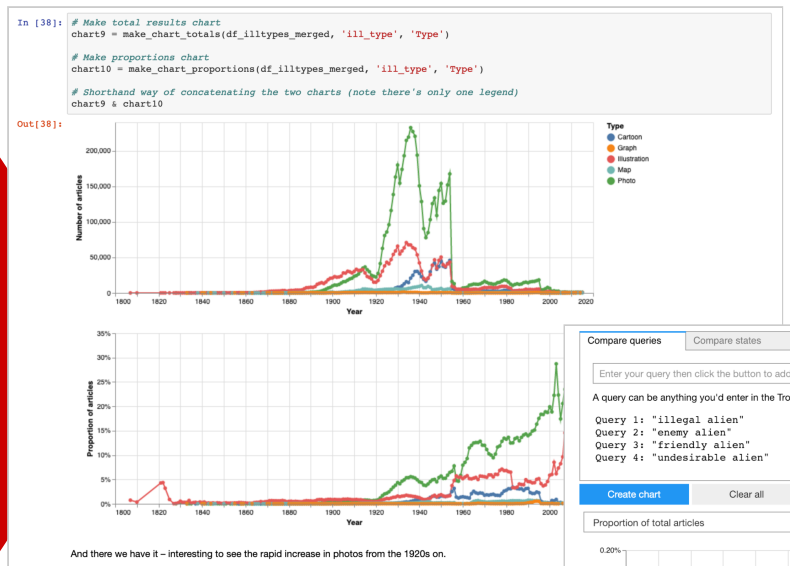
https://glam-workbench.net/trove-journals/

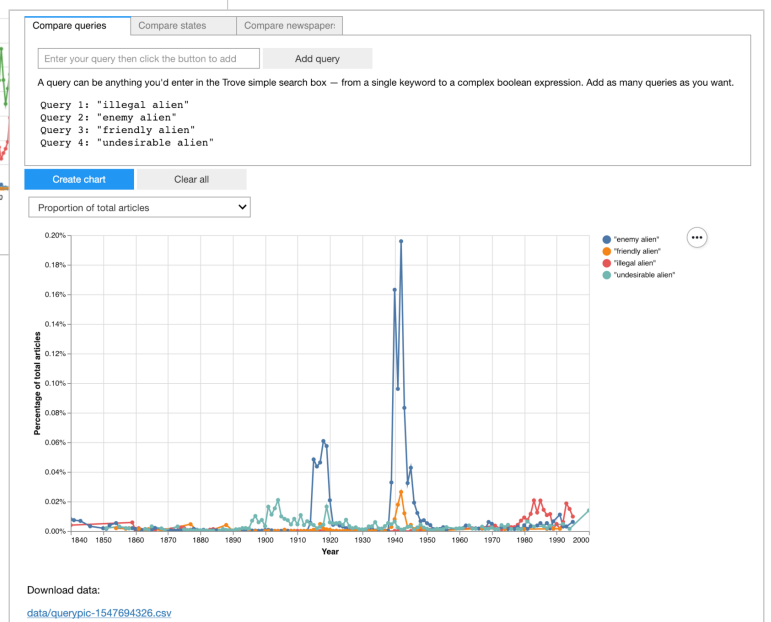
ASKING DIFFERENT QUESTIONS

VISUALISE SEARCHES

(asking historical questions with search facets)

TROVE NEWSPAPERS



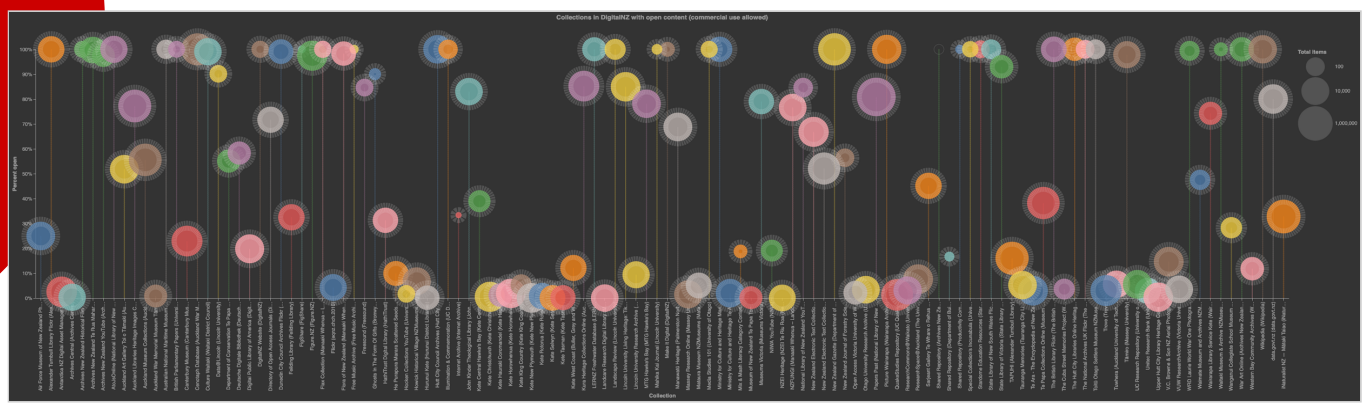


https://glam-workbench.net/trove-newspapers/

OPEN COLLECTIONS

(visualise usage conditions)

DIGITALNZ



https://glam-workbench.net/digitalnz/

COUNTING WORDS

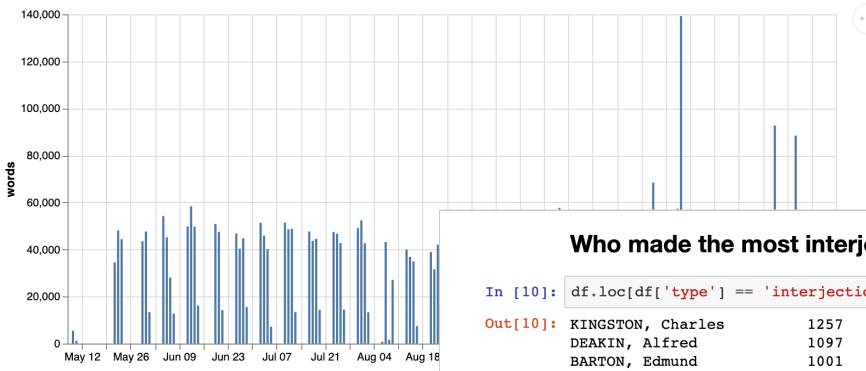
AUSTRALIAN PARLIAMENT

How many words were spoken each day of proceedings?

I've only included words in speeches with identified speakers (including interjections), so some procedural content might not be included in the totals.

In [13]:
```
words_per_day = df.groupby(by=['date'])['words'].sum().to_frame().reset_index()
alt.Chart(words_per_day).mark_bar(size=2).encode(
    x='date:T',
    y='words:Q',
    tooltip=['date:T', 'words:Q']
).properties(width=700)
```

Out[13]:



Who made the most interjections?

In [10]:
```
df.loc[df['type'] == 'interjection']['speaker'].value_counts()[:20]
```

Out[10]:

KINGSTON, Charles	1257
DEAKIN, Alfred	1097
BARTON, Edmund	1001
TURNER, George	906
REID, George	801
MCMILLAN, William	775
MAUGER, Samuel	604
LYNE, William	551
WATSON, John Christian	550
COOK, Joseph	536
HIGGINS, Henry	535
ISAACS, Isaac	482
MCEACHARN, Malcolm	429
THOMSON, Dugald	391
CONROY, Alfred	355
MCCAY, James	355
FORREST, John	332
SOLOMON, Vaiben	321
POYNTON, Alexander	300
MCDONALD, Charles	284

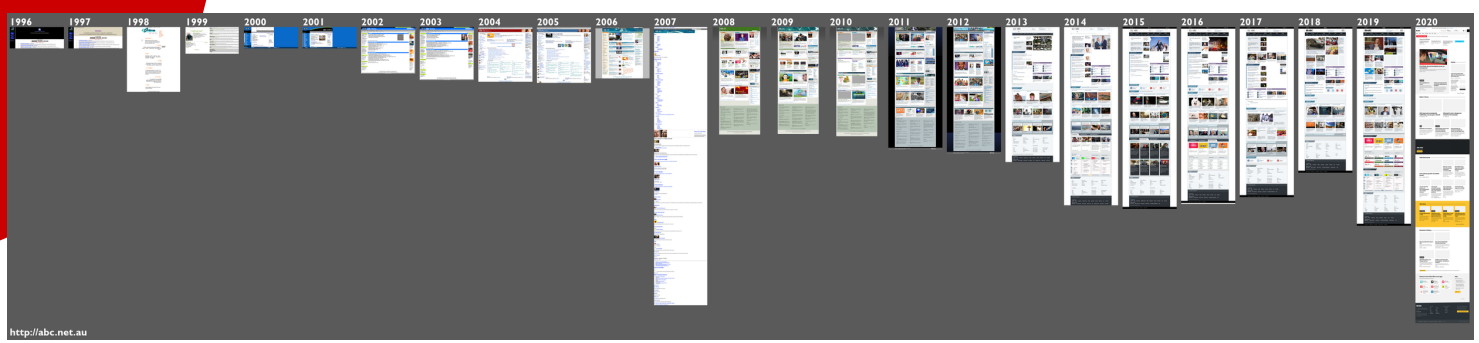
Name: speaker, dtype: int64

https://glam-workbench.net/hansard/

SEEING CHANGE

(screenshots over time)

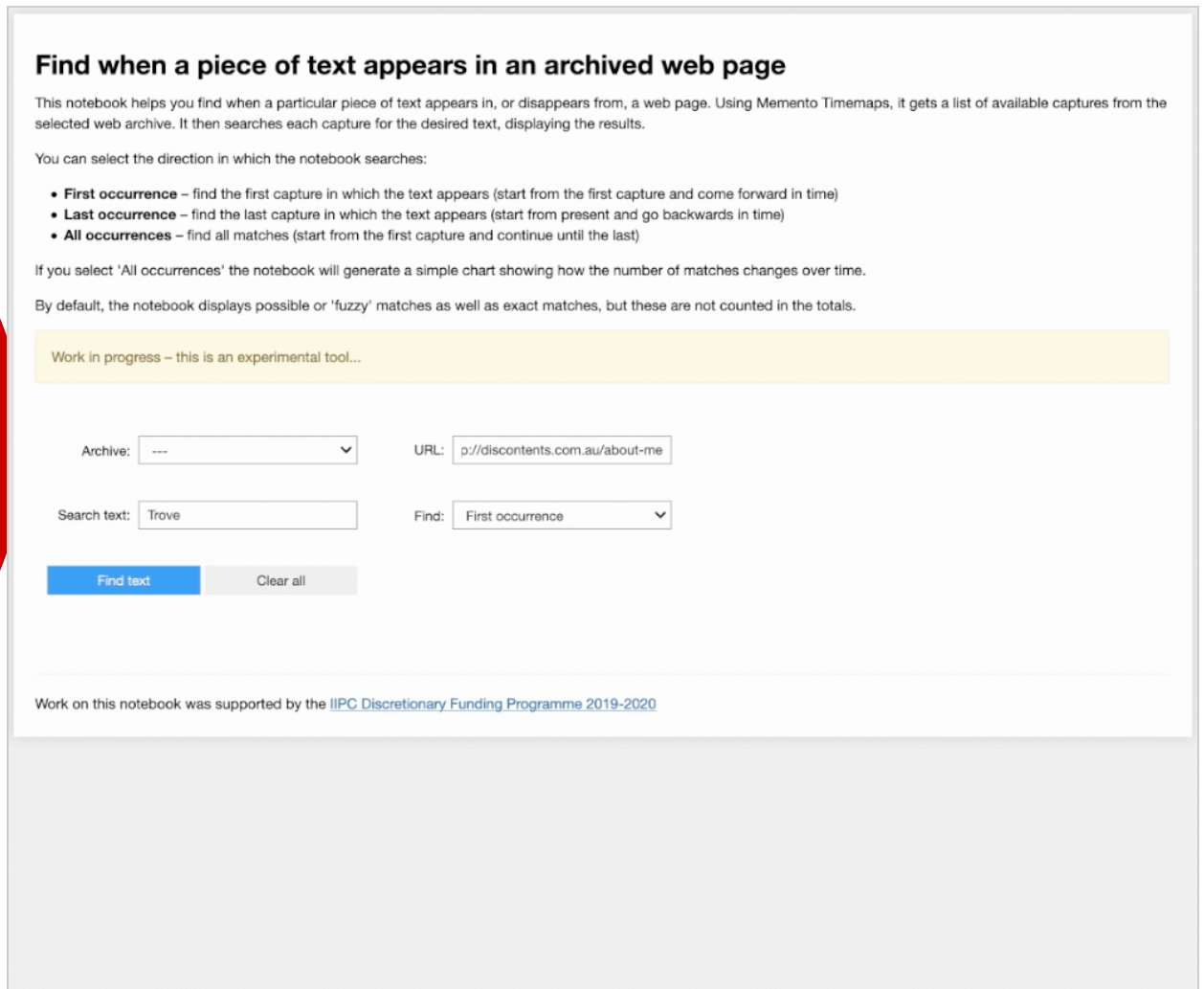
WEB ARCHIVES



https://glam-workbench.net/web-archives/

TRACKING TEXT

WEB ARCHIVES



https://glam-workbench.net/web-archives/

HACKING HERITAGE

EXTEND APIS

(get randomly selected collection items)

RANDOM DIGITALNZ ITEMS

A random item from a specific content partner

[227]:
```
# Get a record
record = get_random_record(content_partner='Puke Ariki')

# Display the results
display(HTML(f'<h4>{record["title"]}</h4>'))
if 'thumbnail_url' in record and record['thumbnail_url']:
    display(Image(url=record['thumbnail_url'], format='jpg'))
display(HTML(f'<p>{record["description"]}</p>'))
display(HTML(f'<a href="{record["landing_url"]}">More...</a>'))
```

```
Additional filters:
  * None
```

St Josephs Parish, Exterior

Exterior of a building and grounds.; Black and White 120 Roll Film/Black and White Negative/Photographic Negative

More...

https://glam-workbench.net/digitalnz/

FIND LANGUAGES

(non-English language newspapers)

TROVE NEWSPAPERS

1. A Voz de Timor (Dili, East Timor : 1970 - 1975)

Language	Language code	Proportion of sample
Portuguese	pt	1.0

2. Adelaider Deutsche Zeitung (SA : 1851 - 1862)

Language	Language code	Proportion of sample
German	de	1.0

3. Australische Zeitung (Adelaide, SA : 1875 - 1916)

Language	Language code	Proportion of sample
German	de	1.0

4. Berita Repoeblik (Djakarta, Indonesia : 1945 - 1946)

Language	Language code	Proportion of sample
Malay (macrolanguage)	ms	0.8913043478260869
Indonesian	id	0.10869565217391304

5. Chinese Republic News (Sydney, NSW : 1914 - 1937)

Language	Language code	Proportion of sample
Chinese	zh	0.9456521739130435

29. Le Courrier Australien (Sydney, NSW : 1892 - 2011)

Language	Language code	Proportion of sample
French	fr	0.8163265306122449
English	en	0.17346938775510204

30. Mediterranean Voice (Perth, WA : 1971 - 1972)

Language	Language code	Proportion of sample
Modern Greek (1453-)	el	0.375
English	en	0.28125
Portuguese	pt	0.10416666666666667
French	fr	0.0625
Spanish	es	0.052083333333333336

31. Meie Kodu = Our Home (Sydney, NSW : 1949 - 1956)

Language	Language code	Proportion of sample
Estonian	et	1.0

32. Mūsų Pastogė = Our Haven (Sydney, NSW : 1950 - 1954)

Language	Language code	Proportion of sample
Lithuanian	lt	0.95

https://glam-workbench.net/trove-newspapers/

PLAY WITH COLLECTIONS

(creating scissors & paste messages)

WORDS FROM OCR

https://glam-workbench.net/trove-newspapers/

BRINGING DOCUMENTATION ALIVE

AN API EXAMPLE

MUSEUMS VICTORIA

A random item from the Museum of Victoria

```
[67]: import requests
      import random
      from IPython.display import Image, display, HTML
      import ipywidgets as widgets

[36]: SEARCH_URL = 'https://collections.museumsvictoria.com.au/api/search'

[72]: def display_random(b):
          out.clear_output()
          params = {
              'query': ' ',
              'sort': 'random',
              'hasimages': 'yes'
          }
          response = requests.get(SEARCH_URL, params=params)
          record = random.choice(response.json())
          with out:
              display(HTML(f'<h3>{record["displayTitle"]}</h3>'))
              display(Image(url=record['media'][0]['small']['uri']))
              display(HTML(f'<a href="https://collections.museumsvictoria.com.au/{record["id"]}">More info...</a>'))

      go = widgets.Button(description='Randomise!')
      out = widgets.Output()
      go.on_click(display_random)
      display(go)
      display(out)
```

Randomise!

Proof Coin - 1/2 Cent, British Caribbean Territories, 1955

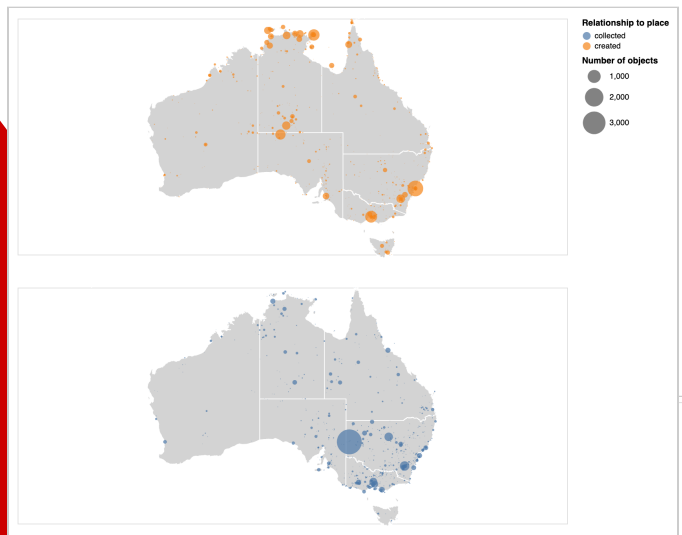
More info...

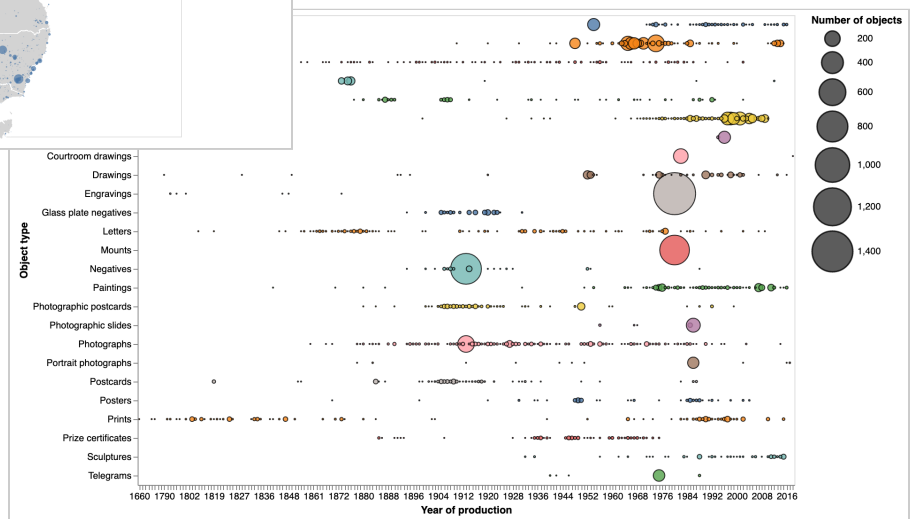
https://glam-workbench.net/museumsvictoria/

EXPLORE AN API

(collections in time & space)

NATIONAL MUSEUM OF AUSTRALIA





https://glam-workbench.net/nma/

DOCUMENT DATA SOURCES

WEB ARCHIVE APIS

Timegates

Timegates let you query a web archive for the capture closest to a specific date. You do this by supplying your target date as the `Accept-Datetime` value in the headers of your request.

For example, if you wanted to query the Australian Web Archive to find the version of `http://nla.gov.au/` that was captured as close as possible to 1 January 2001, you'd set the `Accept-Datetime` header to header to 'Fri, 01 Jan 2010 01:00:00 GMT' and request the url:

```
https://web.archive.org.au/awa/http://nla.gov.au/
```

A `get` request will return the captured page, but if all you want is the url of the archived page you can use a `head` request and extract the information you need from the response headers. Try this:

```
In [3]: response = requests.head('https://web.archive.org.au/awa/http://nla.gov.au/', headers={'Accept-Datetime': 'Fri, 01 Jan 20
        response.headers

Out[3]: {'Server': 'nginx', 'Date': 'Fri, 22 May 2020 02:40:23 GMT', 'Content-Length': '0', 'Connection': 'keep-alive', 'L
```

The request above returns the following headers:

```
{
    'Server': 'nginx',
    'Date': 'Wed, 06 May 2020 04:34:50 GMT',
    'Content-Length': '0', 'Connection': 'keep-alive',
    'Location': 'https://web.archive.org.au/awa/20100205144227/http://nla.gov.au/',
    'Link': '<http://nla.gov.au/>; rel="original", <https://web.archive.org.au/awa/http://nla.gov.au/>; rel="timega
    'Vary': 'accept-datetime'
}
```

The `Link` parameter contains the Memento information. You can see that it's actually providing information on four types of link:

- the `original` url (ie the url that was archived) – `<http://nla.gov.au/>`
- the `timegate` for the harvested url (which us what we just used) – `<https://web.archive.org.au/awa/http://nla.gov.au/>`
- the `timemap` for the harvested url (we'll look at this below) – `<https://web.archive.org.au/awa/timemap/link/http://nla.gov.au/>`
- the `memento` – `<https://web.archive.org.au/awa/20100205144227mp_/http://nla.gov.au/>`

The `memento` link is the capture closest in time to the date we requested. In this case there's only about a month's difference, but of course this will depend on how frequently a url is captured. Opening the link will display the capture in the web archive. As we'll see below, some systems provide additional links such as `first memento`, `last memento`, `prev memento`, and `next memento`.

https://glam-workbench.net/web-archives/

PATHWAYS

BUILDING CONFIDENCE

Out[11]:

	keyword	total
0	cat	93
1	dog	143
2	kangaroo	791
3	koala	246

Our dataset is tiny, so it's easy to see what's going on. If you have lots of data, Pandas can help you make sense of it. For example, we might want to find the keyword with the highest number of results.

Let's try visualising our dataset

We've displayed our data as a table, but a chart would be easier to interpret at a glance. There are a number of charting and data visualisation packages available for Python, here we'll be using Altair. You just feed Altair a dataframe, and tell it the columns to display on each axis. Let's start with a simple bar chart that shows the keywords along the `x` axis, and the number of search results on the `y` axis.

In [ ]:
```
import altair as alt

# If you're using Jupyter Lab rather than Jupyter Notebook,
# change 'notebook' to 'default' in the line below
alt.renderers.enable('notebook')

alt.Chart(df).mark_bar().encode(
    x='keyword:N',
    y='total:Q'
)
```

Help! I get a weird error saying 'require is not defined', or a message saying something about 'Vegalite'.

If you're using Altair in Jupyter Lab rather than Jupyter Notebook, you need to make an adjustment to the code above. Just change 'notebook' to 'default' in the line starting 'alt.renderers.enable' and run the cell again.

Altair is easy to customise. Here's a few things you could try:

- Switch the `x` and `y` values in the code above and see what happens.
- Change `mark_bar` to `mark_line`.

We'd better save our dataset for later

Many of the notebooks in the GLAM Workbench help you harvest data from GLAM collections, just as we did above. Once you've created a new dataset, you'll probably want to save it. Pandas makes it easy to save your dataframe as a CSV (Comma Separated Values) file. CSV files are simple text files that can be opened by any spreadsheet program. They're widely used for storing and sharing datasets.

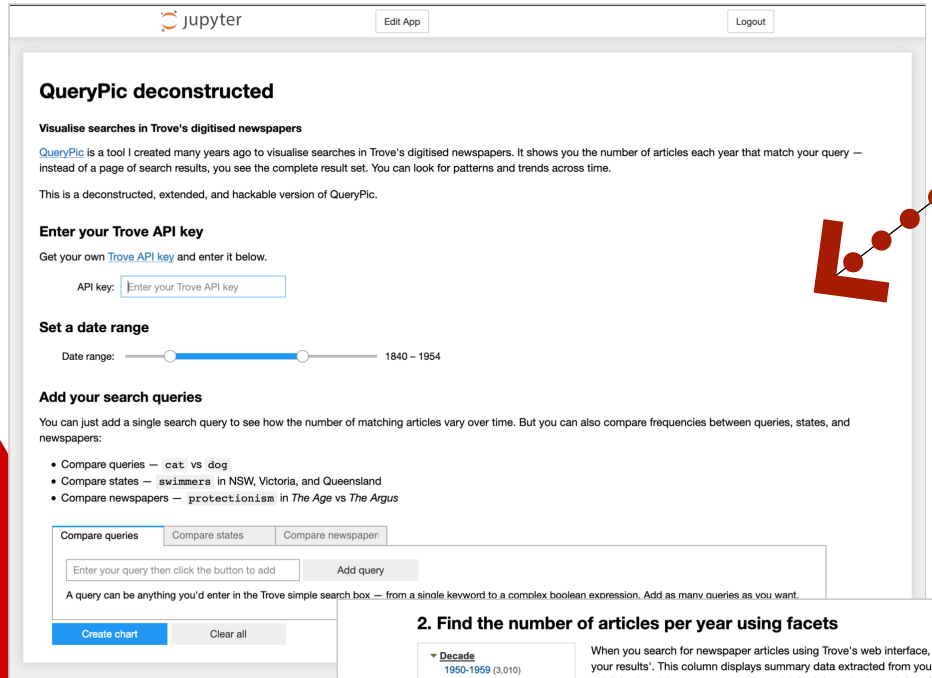
In [ ]:
```
# RUN THIS CELL to save your dataset as a CSV file
```

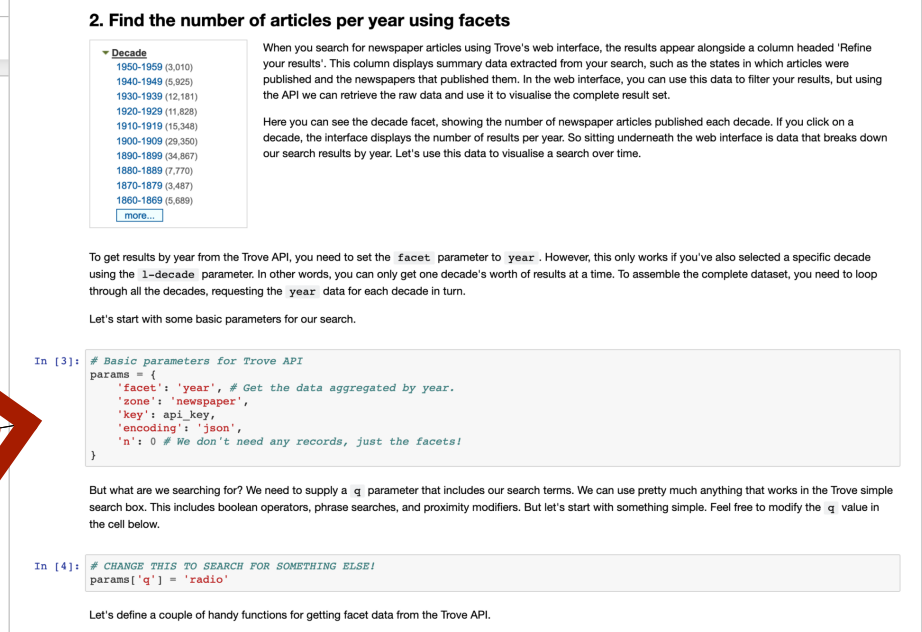
https://glam-workbench.net/getting-started/

BUILDING CONFIDENCE

APP

NOTEBOOK





RUNNING JUPYTER NOTEBOOKS

BINDER IN GLAM WORKBENCH

RUN LIVE IN BINDER

BINDER

Save a Trove newspaper article as an image ¶

Sometimes you want to be able to save a Trove newspaper article as an image. Unfortunately, the Trove web interface doesn't make this easy. The 'Download JPG' option actually loads an HTML page, and while you could individually save the images embedded in the HTML page, often articles are sliced up in ways that make the whole thing hard to read and use. This notebook grabs the page on which an article was published, and then crops the page image to the boundaries of the article. The result is a complete, intact image which presents the article as it was originally published. And if the article is split across multiple pages, you'll get one image per page.

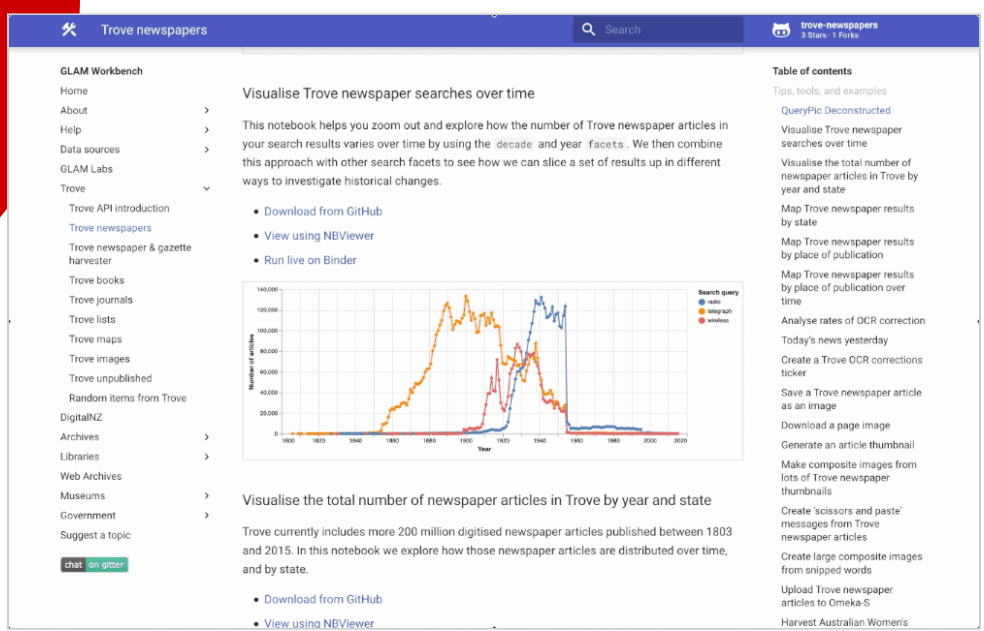
- Download from GitHub
- View using NBViewer
- Run live on Binder
- Run as an app using Voila (the easiest, no code option!)

https://glam-workbench.net/using-binder/

BINDER IN GLAM WORKBENCH

THE MAGIC OF BINDER

- a single click to start
- batteries included (no software for the user to install)
- encourages experimentation
- **just try it...**



RECLAIM CLOUD

- one-click installation
- persistent environmments
- low cost

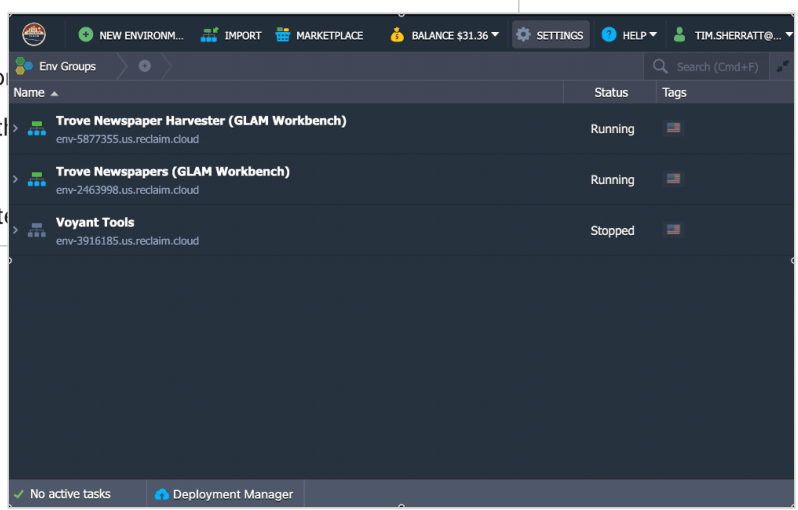
MORE WAYS TO RUN

Using Reclaim Cloud ¶

Launch on Reclaim Cloud

Reclaim Cloud is a paid hosting service, aimed particularly at supported digital scholarship in hte humanities. Unlike Binder, the environments you create on Reclaim Cloud will save your data – even if you switch them off! To run this repository on Reclaim Cloud for the first time:

- Create a Reclaim Cloud account and log in.
- Click on the button above to start the installatio
- A dialogue box will ask you to set a password, t installation.
- Sit back and wait for the installation to complet



https://glam-workbench.net/using-reclaim-cloud/

DOCKER

- run locally
- persistent environments
- free, but more knowledge needed

MORE WAYS TO RUN

Quick start

The GLAM Workbench repositories are stored as pre-built 'Images' on Docker Hub. To download and run one of these images for the first time, you need to:

- Install Docker Desktop.
- Create a new directory to contain your local files, and open it from the command line. This directory will be named `work` in the Jupyter interface.
- From the command line, run the following command, replacing `[REPOSITORY NAME]` with the name of a GLAM Workbench repository, for example, 'trove-newspapers':

```
docker run -p 8888:8888 --name [REPOSITORY NAME] -v "$PWD":/home/jovyan/work gl
```

- It will take a while to download and configure the Docker image. Once it's ready you'll see a message saying that Jupyter Notebook is running.
- Point your web browser to `http://127.0.0.1:8888`
- To stop the container hit `^ Ctrl` + `C`, or use one of the methods described below.

https://glam-workbench.net/using-docker/

GLAM WORKBENCH IS OPEN!

COPY!
EDIT!
RE-USE!

- free!
- openly licensed to encourage reuse & modification
- shared through GitHub
- preserved in Zenodo
- use it, share it, change it

CONTACT ME

@wragge on Twitter

timsherratt.org

GLAM Workbench issues on GitHub

GLAM Workbench on OzGLAM Help

Don't have a question? Why not ask about my t-shirt?...