

Linked Art

Webinar: Linked Art in Practice
using Jupyter Code Notebooks

*Connecting Cultural
Heritage Collections*

Tanya Gray

tanya.gray@humanities.ox.ac.uk



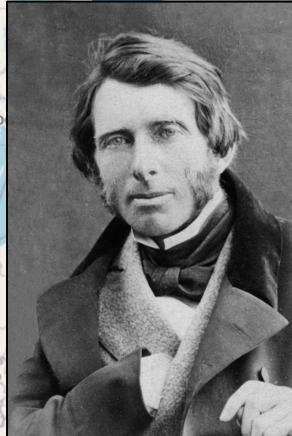
- **Linked Art**
- **Data Visualisations**
- **Code notebooks**



[Map Overview](#)

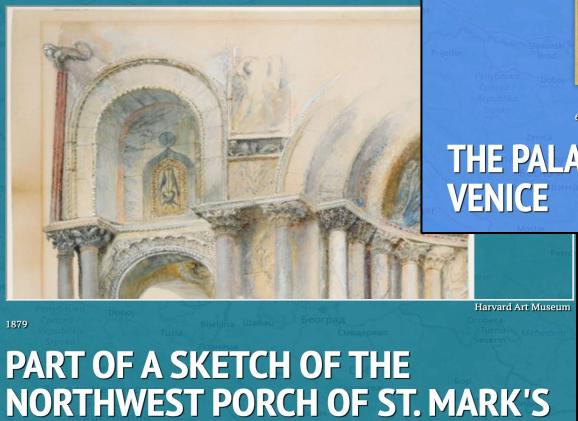


**THE PALAZZO CONTARINI-FASAN,
VENICE**



BOAT AND SKETCHES OF TWO FIGURES, VENICE

knight lab



Code Notebooks

http://localhost:8888/notebooks/01-06-Transform-John-Ruskin.ipynb

File Edit View Insert Cell Kernel Widgets Help

Load NGA Collection Data into DataFrame

```
In [31]: try:
    import pandas as pd
except:
    %pip install pandas
    import pandas as pd

fileNGA = 'data/nga/input/nga_ruskin.csv'

dataFrameNGA = pd.read_csv(fileNGA)
dataFrameNGA.head()
```

Out[31]:

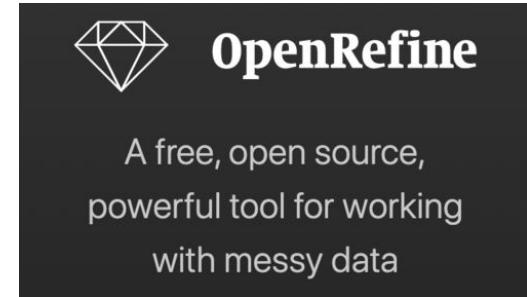
	objectid	accessioned	accessionnum	locationid	title	displaydate	beginyear	endyear	visualbrowserurl
0	70238	1	1987.73.2	NaN	Tower of the Cathedral at Sens	c. 1845	1845	1845	1826
1	70367	1	1988.20.38	NaN	Tree Study	mid-1850s	1845	1855	1826



Notebook	Download	nbviewer	Binder
Anapolis Museum of Art	download	nbviewer	launch binder
Philadelphia Museum of Art	download	nbviewer	launch binder
Welland Museum of Art	download	nbviewer	launch binder
Welland Museum of Art - simplified	download	nbviewer	launch binder
National Gallery of Art	download	nbviewer	launch binder
Transform	Harvard Art Museum	download	nbviewer
Transform	Rijksmuseum	download	nbviewer
Transform	Ashmolean Museum	download	nbviewer
Transform	John Ruskin artworks - Transform Data	download	nbviewer
Reconcile	John Ruskin artworks - Reconcile place names	download	nbviewer
Visualise	John Ruskin artworks - Timeline	download	nbviewer
Visualise	John Ruskin artworks - StoryMap	download	nbviewer



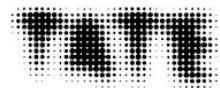
Visual Studio Code



Data Visualisations



ASHMOLEAN
MUSEUM
OXFORD



Harvard
Art Museum

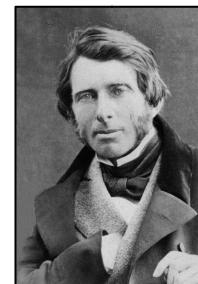
Philadelphia
Museum of
Art

RIJKSMUSEUM
amsterdam

RIJKS MUSEUM

Museum & Galleries
Collection Data

John Ruskin. artist

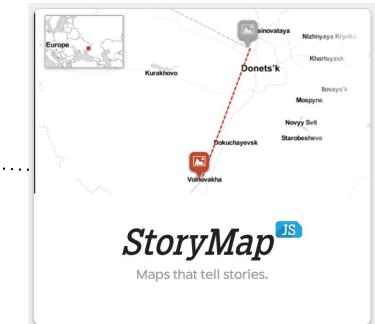


Unified representation with
Linked Art

Data visualisation service

StoryMap^{JS}

Maps that tell stories.



Timeline^{JS}

Easy-to-make, beautiful timelines.

Linked Art:
Sustainable Cultural Knowledge
through Linked Open Usable Data

<https://linked.art/>

knight lab

John Ruskin

- Prolific artist
- Social commentator
- European travels
- Depicted nature and architecture
- Influential for ideas on society, art, craft, architecture, building preservation

Artworks now in many private and public collections

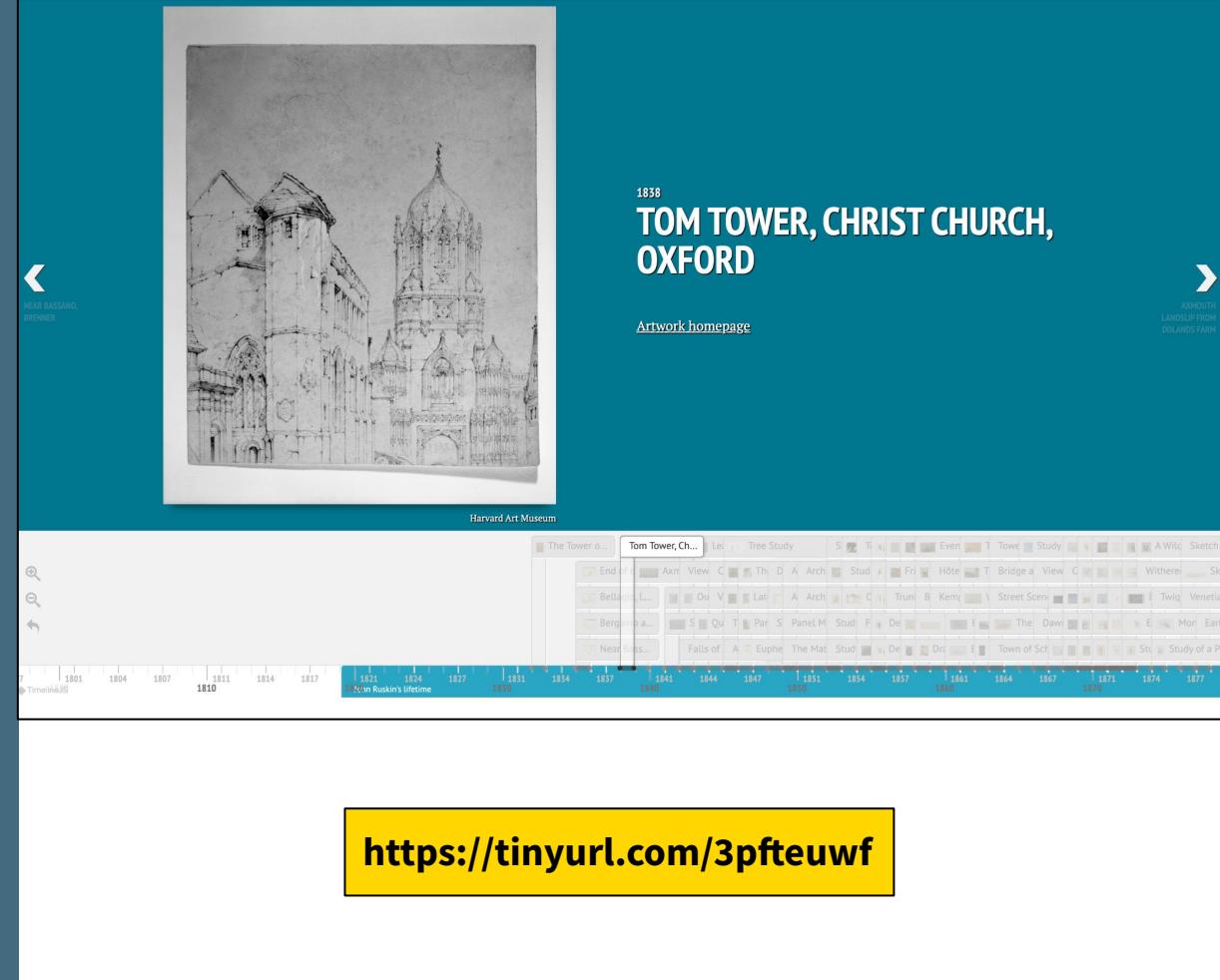


John Ruskin

Timeline visualisation

Uses

- Museum and Gallery collection data
- unified representation with Linked Art
- KnightLab Vis



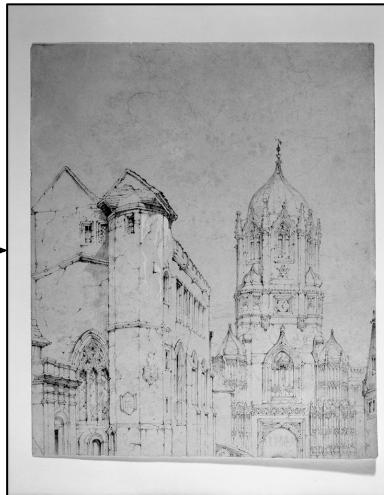
John Ruskin Timeline Visualisation

The image shows a digital exhibition interface. At the top, there is a painting titled "VIEW OF CHAMONIX" by J.M.W. Turner. Below the painting is a color calibration strip. To the left of the painting is a left arrow icon, and to the right is a right arrow icon. To the right of the painting, the title "VIEW OF CHAMONIX" is displayed in large white letters, with "J.M.W. TURNER" above it. Below the title is the caption "An early view of the valley". At the bottom of the interface is a horizontal timeline. The timeline has numerical markers at 1820, 1830, 1840, 1850, 1860, 1870, 1880, 1890, and 1900. A blue bar highlights the year 1866. A tooltip or dropdown menu is visible over the timeline, containing the text "At the end of 1866, Ruskin had just completed his first major work on architecture, 'The Seven Lamps of Architecture'." The background of the interface is teal.

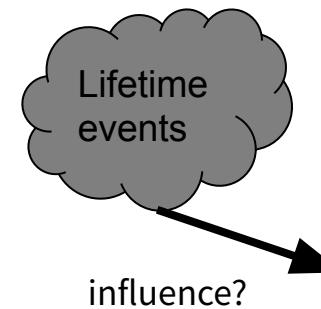
Timeline - Future Scholarship



Early influence



Early artwork



Later artwork

Samuel Prout, Artist

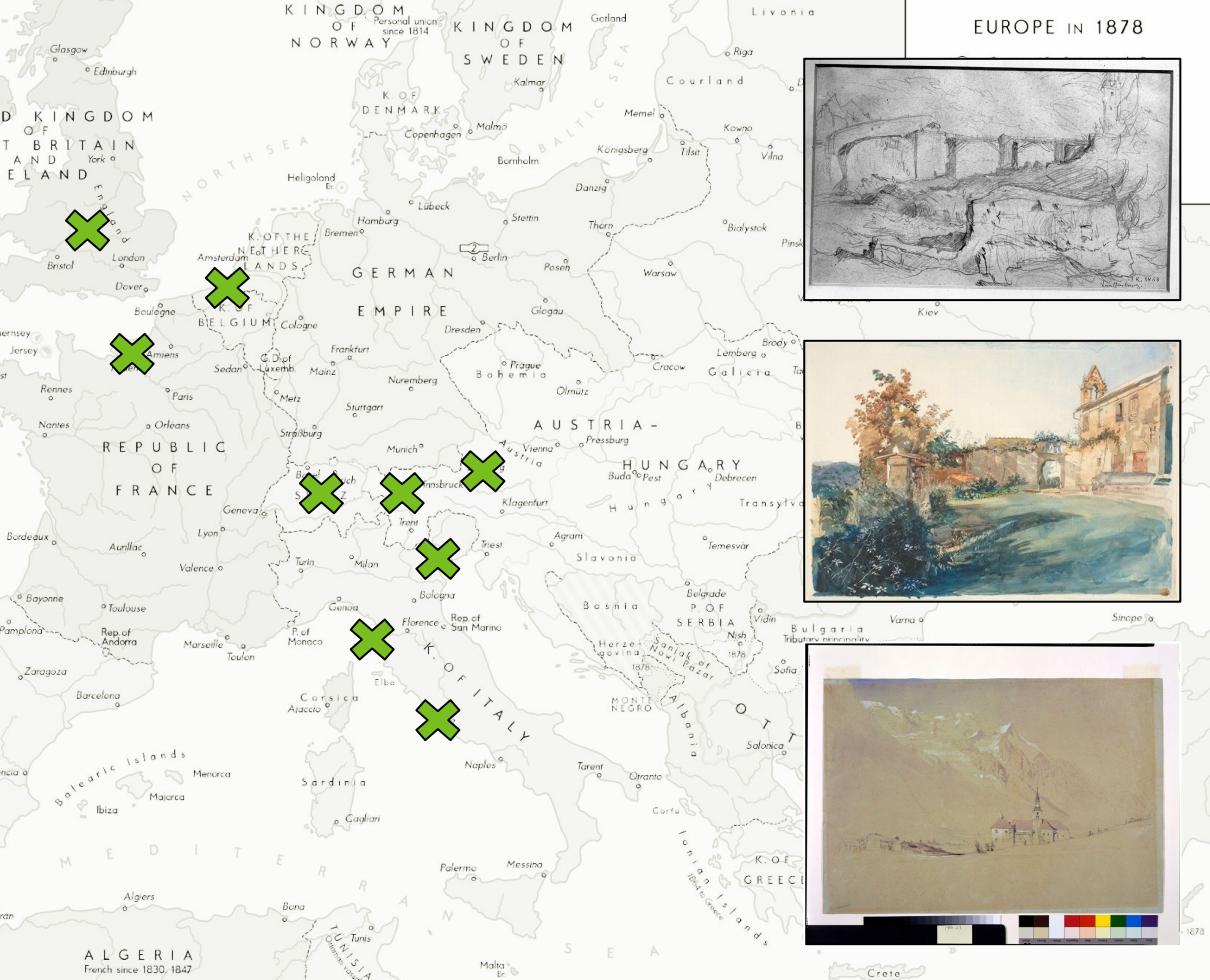
A change in characteristics of Ruskin's artwork through time?

John Ruskin

Travel

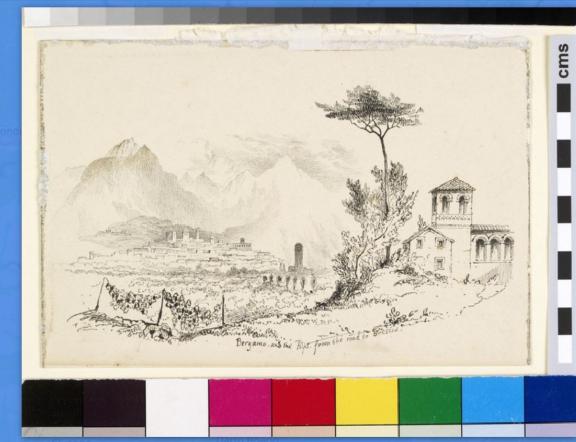
Extensive travel in Europe over his lifetime, often to Italy

Recorded travels with drawings and paintings of natural scenery and buildings



John Ruskin

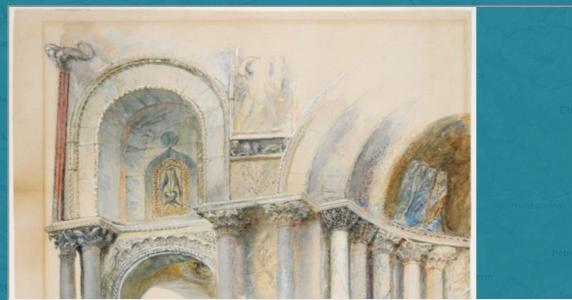
Place name in title



1855

Ashmolean Museum

BELLAGIO, LAGO DI COMO



1879

Harvard Art Museum

PART OF A SKETCH OF THE
NORTHWEST PORCH OF ST. MARK'S



October 1874

Ashmolean Museum

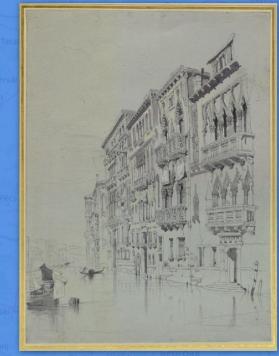
MONT BLANC FROM SAINT-MARTIN-SUR-ARVE



1855

Ashmolean Museum

BERGAMO
ROAD TO B



6-16 May 1841

Ashmolean Museum

THE PALAZZO CONTARINI-FASAN,
VENICE

John Ruskin

StoryMap

Uses:

- Collection data
- Unified with Linked Art
- Reconciled with Getty Thesaurus of Geographic Names to extract geocoordinates
- KnightLab vis

<https://tinyurl.com/mrxwv3um>

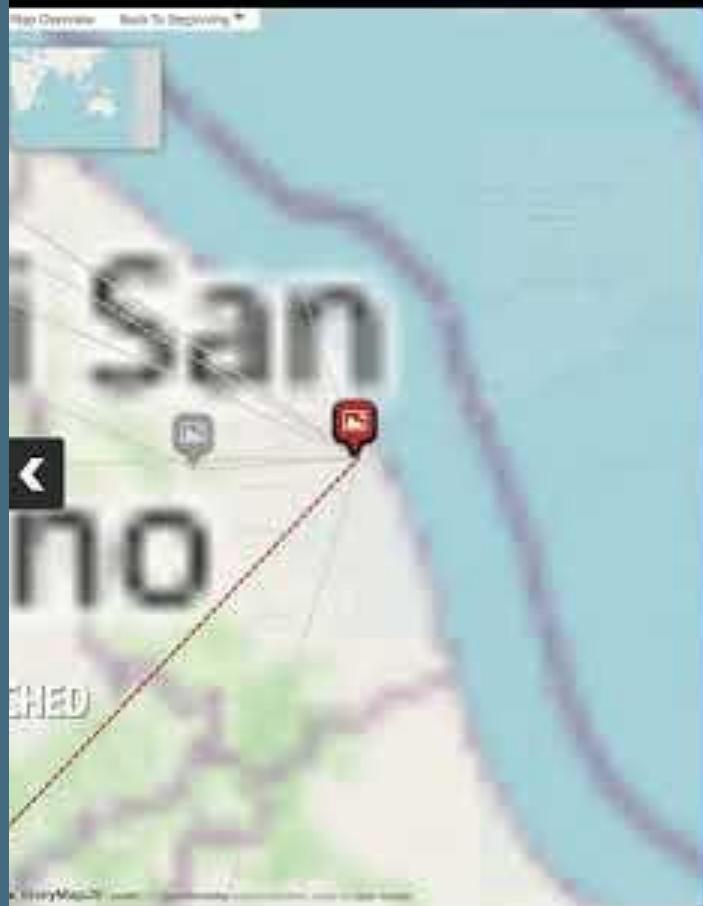
Map Overview Back To Beginning

United Kingdom
Éire / Ireland
Danmark
Hamburg
Berlin
Polška
Česko
Slovensko
Magyarország
Zagreb
Србија
Bosna i Hercegovina
Hrvatska
Italia
Portugal
España
Barcelona
Rabat
الرباط
Algiers
الجزائر
Tunis
تونس
Tripoli
طرابلس
Casablanca
الدار البيضاء
Algiers
الجزائر
Maroc / مملكة المغرب
Algérie / الجزائر
Libya / Libya
Pohjanlahti - Bottniska viken
Sverige
Oslo
Stockholm
Helsinki
Eesti
Latvia
Lietuva
Baltija
Baltic Sea
Golf de Gascogne / Golfo de Vizcaya
Golfo de Vizcaya
Paris
France
1845
Harvard Art Museum
BOAT AND SKETCHES OF TWO FIGURES, VENICE
Fine Arts Department, Harvard University, Cambridge, MA, Transferred to the Fogg Art Museum, 1926.
Artwork homepage

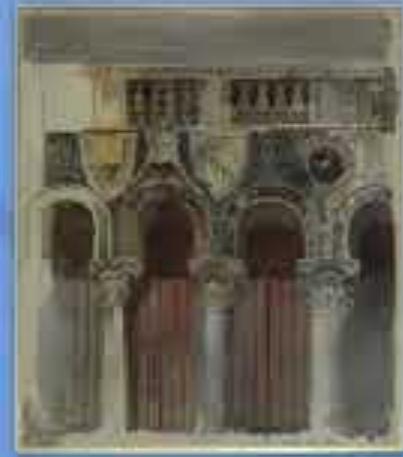
StoryMapJS | Leaflet | © OpenStreetMap and contributors, under an open license.

John Ruskin

StoryMap



STUDY OF THE MARBLE INLAYING
ON THE FRONT OF THE CASA
LOREDAN, VENICE



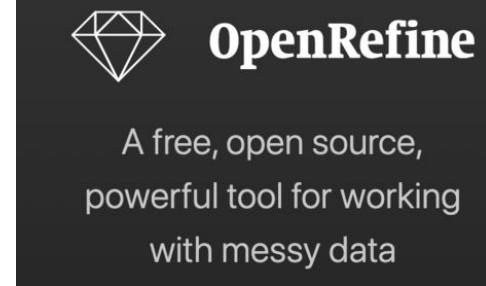
Code Notebooks



ANACONDA®



Visual Studio Code



Code Notebooks

https://github.com/tgra/Linked-Art

README.md

Notebook type	Notebook	Download	nbviewer	Binder
Transform	Indianapolis Museum of Art	download	nbviewer	launch binder
Transform	Philadelphia Museum of Art	download	nbviewer	launch binder
Transform	Cleveland Museum of Art	download	nbviewer	launch binder
Transform	Cleveland Museum of Art - simplified	download	nbviewer	launch binder
Transform	National Gallery of Art	download	nbviewer	launch binder
Transform	Harvard Art Museum	download	nbviewer	launch binder
Transform	Rijksmuseum	download	nbviewer	launch binder
Transform	Ashmolean Museum	download	nbviewer	launch binder
Transform	John Ruskin artworks - Transform Data	download	nbviewer	launch binder
Reconcile	John Ruskin artworks - Reconcile place names	download	nbviewer	launch binder
Visualise	John Ruskin artworks - Timeline	download	nbviewer	launch binder
Visualise	John Ruskin artworks - StoryMap	download	nbviewer	launch binder

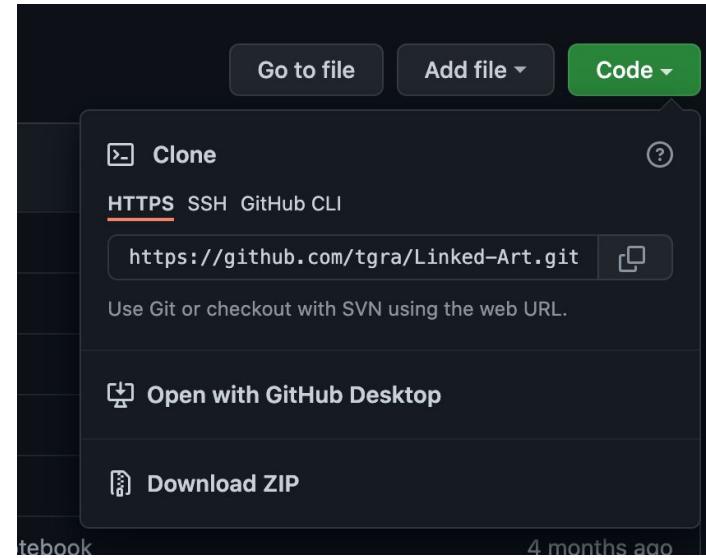
Where to Find Linked Art Code Notebooks

GitHub

github.com/tgra/Linked-Art

Install Git

- Check out with git clone
- Download ZIP



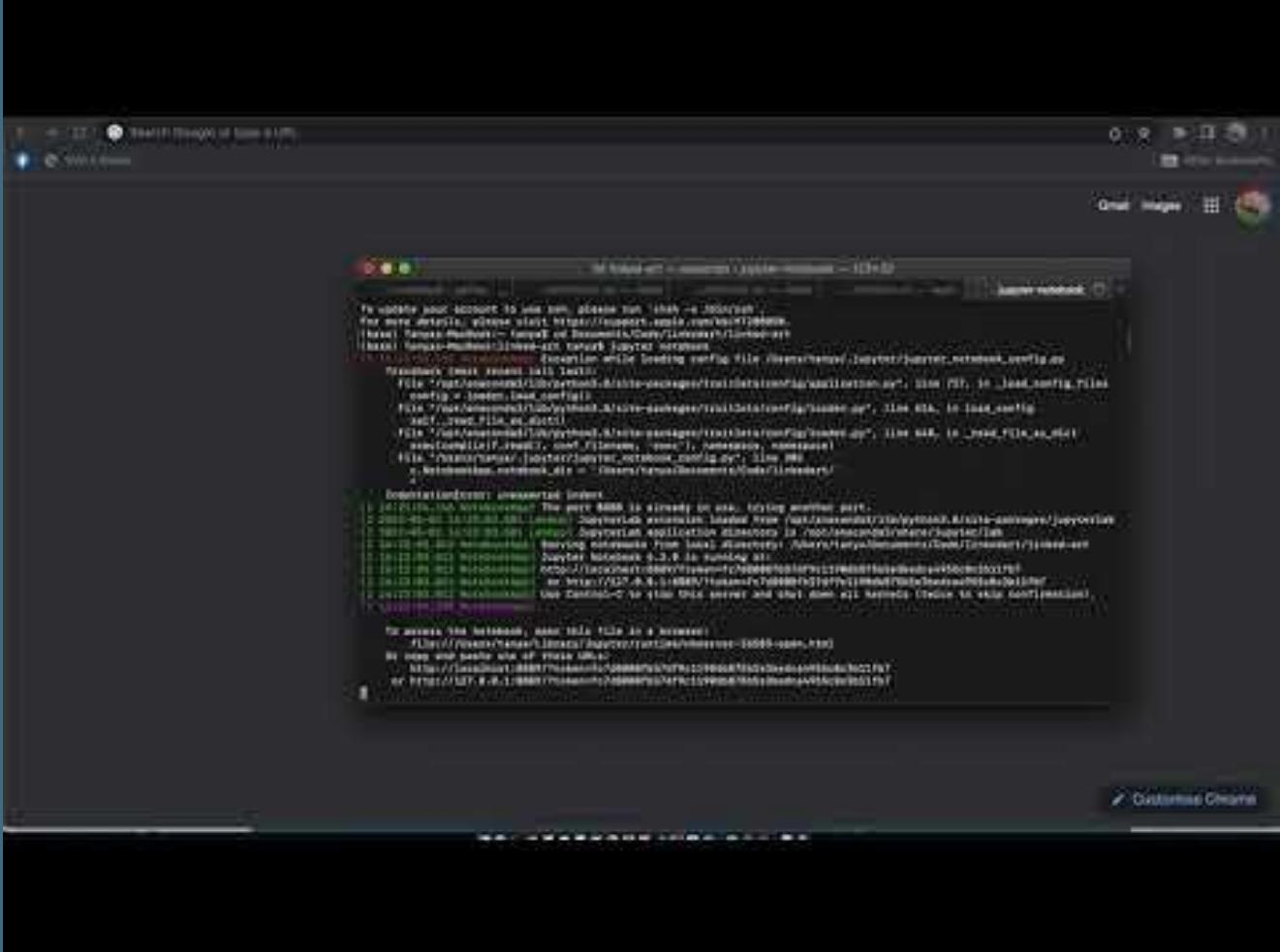
Notebook Tools

- Jupyter command line
- Binder
- Anaconda
- Jupyter Nbviewer
- Visual Studio code
- JupyterHub



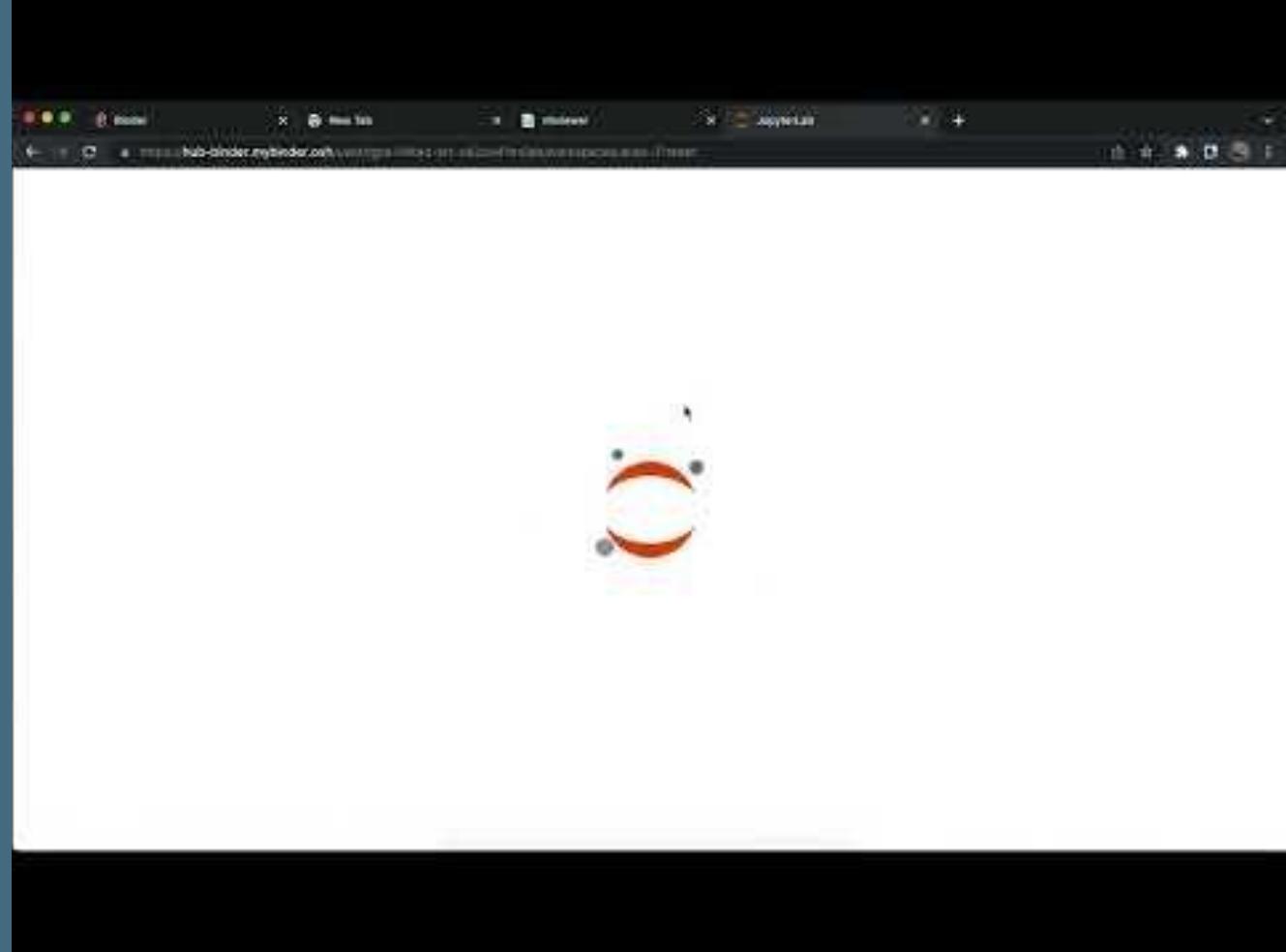
Notebook Tools

Jupyter Notebook via command line



Notebook Tools

Binder via
GitHub



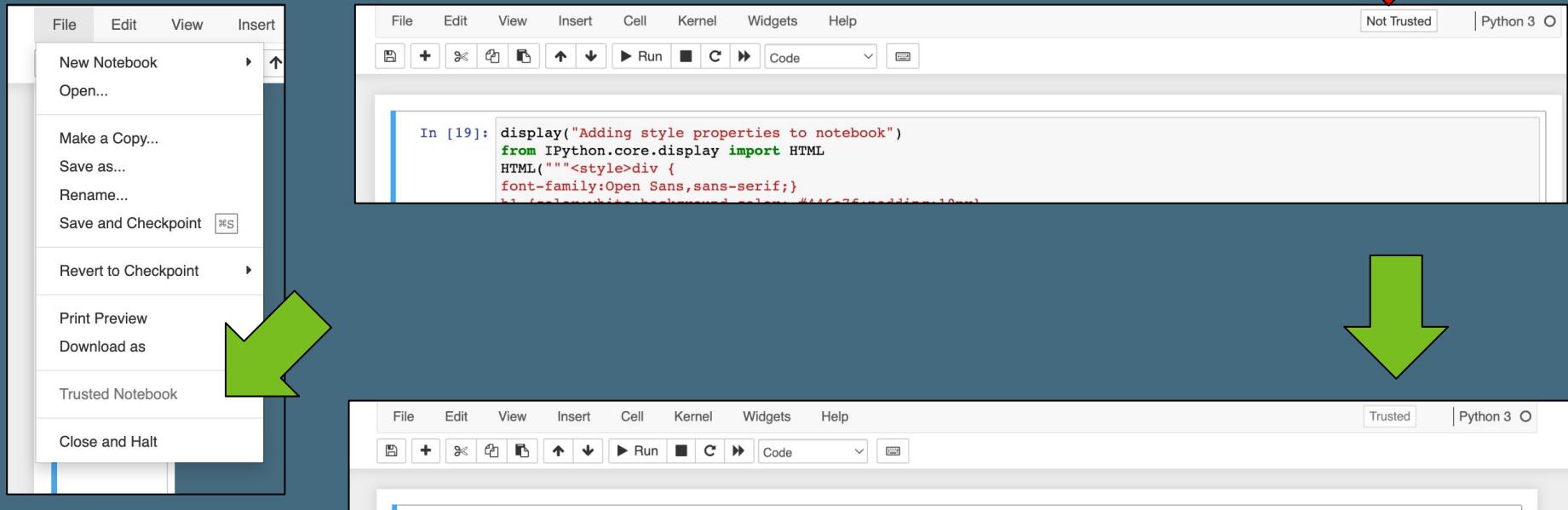
Notebook Tools

Nbviewer

nbviewer.org



Jupyter Notebooks - Trusted vs Not Trusted



How do I trust a notebook?

Users can explicitly trust a notebook in two ways: 1. **At the command-line**, with: `jupyter trust /path/to/notebook.ipynb`. 2. **After loading** the untrusted notebook, with File / Trust Notebook.



Transformation



Reconciliation



Visualisation

Transformation Code Notebook

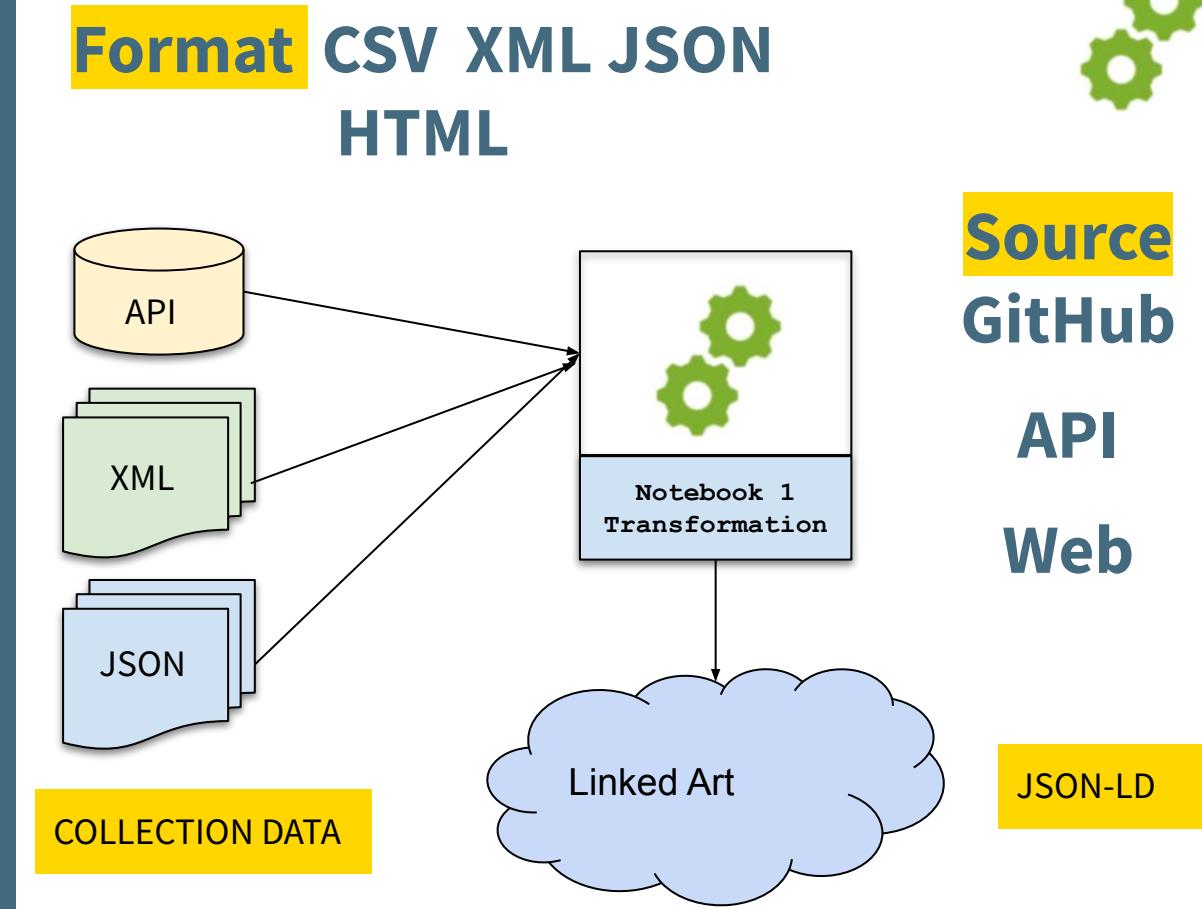


Transformation notebooks demonstrate how to:

- **Extract** data
 - from source
- **Map** data
 - to the Linked Art data model
- **Transform** data
 - to a unified representation using Python code
- **Publish** data
 - as JSON-LD files

Extract Data

- Locate
- Different formats
- Different data models
- Read data into Python dictionary



Code Notebooks

https://github.com/tgra/Linked-Art

README.md

Notebook type	Notebook	Download	nbviewer	Binder
Transform	Indianapolis Museum of Art	download	nbviewer	launch binder
Transform	Philadelphia Museum of Art	download	nbviewer	launch binder
Transform	Cleveland Museum of Art	download	nbviewer	launch binder
Transform	Cleveland Museum of Art - simplified	download	nbviewer	launch binder
Transform	National Gallery of Art	download	nbviewer	launch binder
Transform	Harvard Art Museum	download	nbviewer	launch binder
Transform	Rijksmuseum	download	nbviewer	launch binder
Transform	Ashmolean Museum	download	nbviewer	launch binder
Transform	John Ruskin artworks - Transform Data	download	nbviewer	launch binder
Reconcile	John Ruskin artworks - Reconcile place names	download	nbviewer	launch binder
Visualise	John Ruskin artworks - Timeline	download	nbviewer	launch binder
Visualise	John Ruskin artworks - StoryMap	download	nbviewer	launch binder

Map Data

Understand

- source data model
- Linked Art data model

Manual process

```
mapp = {
    "id": "id",
    "accession_number": "accession_number",
    "accession_date": "",
    "classification": "type",
    "title": "title",
    "alt_title": "title_in_original_language",
    "notes": "tombstone",
    "date_created": "creation_date",
    "date_created_earliest": "creation_date_earliest",
    "date_created_latest": "creation_date_latest",
    "created_period": "culture",
    "created_dynasty": "",
    "created_inscriptions": "inscriptions",
    "created_notes": "fun_fact",
    "creator": "creator",
    "physical_medium": "Medium",
    "physical_style": "",
    "physical_technique": "technique",
    "physical_description": "",
    "physical_dimensions": "measurements",
    "created_provenance": "provenance",
    "credit_line": "creditline",
}
```

id	id
accession_number	accession_number
accession_date	
classification	type
title	title
alt_title	title_in_original_language
notes	tombstone
date_created	creation_date
date_created_earliest	creation_date_earliest
date_created_latest	creation_date_latest
created_period	culture
created_dynasty	
created_inscriptions	inscriptions
created_notes	fun_fact
creator	NaN
physical_medium	Medium
physical_style	
physical_technique	technique
physical_description	
physical_dimensions	measurements
created_provenance	provenance
credit_line	creditline
collection	department



Transform Data

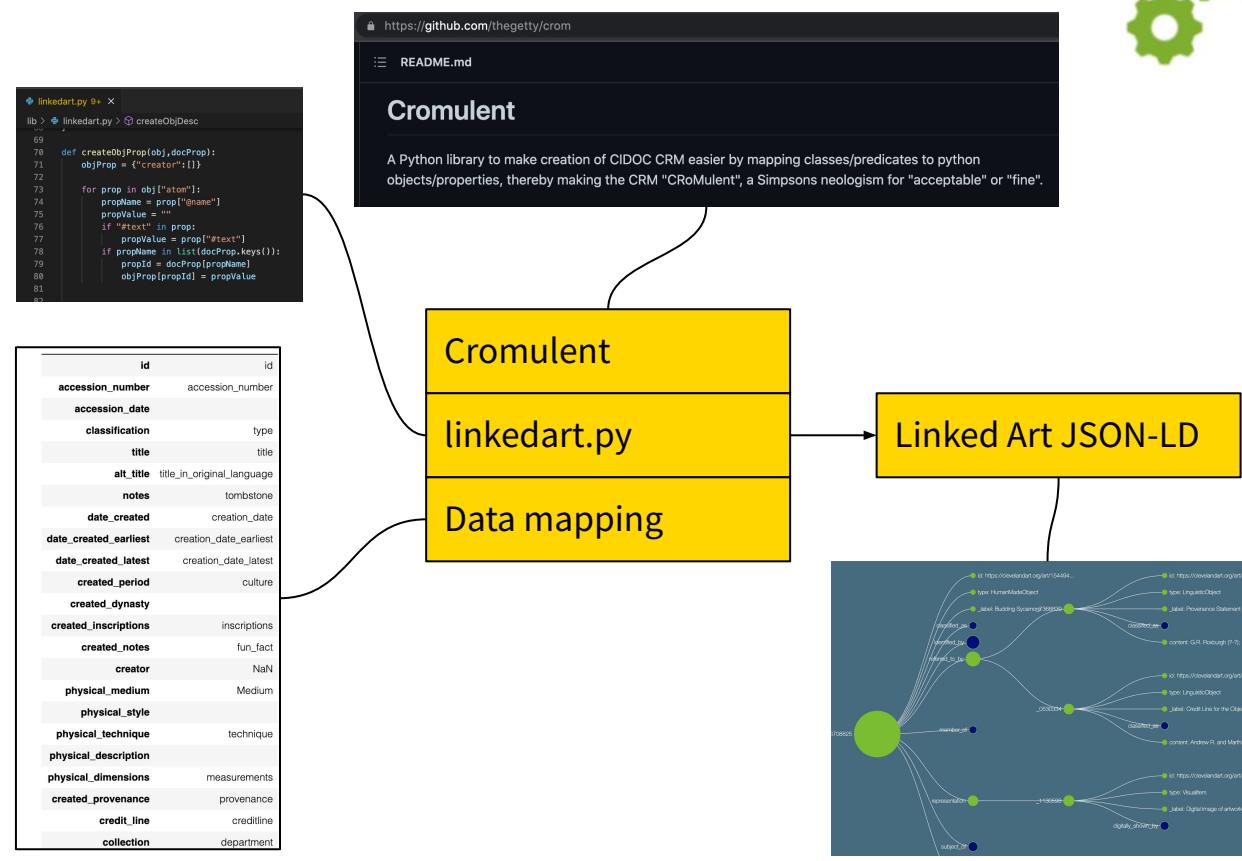


Uses

- Data mapping
- Cromulent
- Custom code linkedart.py

Creates

- Linked Art
- JSON-LD



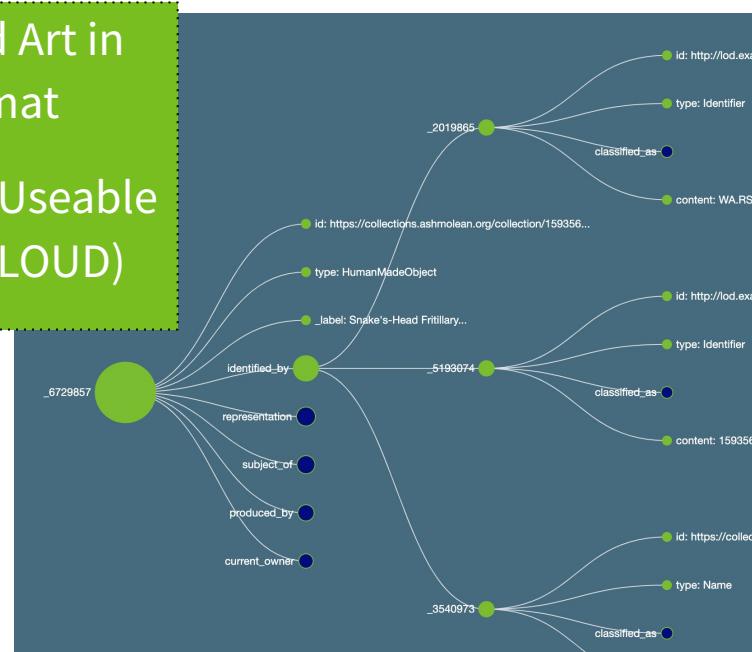
Transformation Sub-steps: Extract → Map → Transform → Publish

Transformation - Publish



Create Linked Art in
JSON-LD format

Linked Open Useable
Data format (LOUD)



```

print(factory.toString(objLA, compact=False))

{
  "@context": "https://linked.art/ns/v1/linked-art.json",
  "id": "https://clevelandart.org/art/74540",
  "type": "HumanMadeObject",
  "label": "Leda and the Swan",
  "classified_as": [
    {
      "id": "http://vocab.getty.edu/aat/300033973",
      "type": "Type",
      "_label": "drawing",
      "classified_as": [
        {
          "id": "http://vocab.getty.edu/aat/300435443",
          "type": "Type",
          "_label": "type of Work"
        }
      ]
    }
  ],
  "identified_by": [
    {
      "id": "http://lod.example.org/museum/Identifier/2015.451",
      "type": "Identifier",
      "classified_as": [
        {
          "id": "http://vocab.getty.edu/aat/300312355",
          "type": "Type",
          "_label": "Accession Number"
        }
      ],
      "content": "2015.451"
    },
    {
      "id": "http://lod.example.org/museum/Identifier/74540",
      "type": "Identifier",
      "classified_as": [
        ...
      ]
    }
  ]
}

```



Transformation



Reconciliation



Visualisation

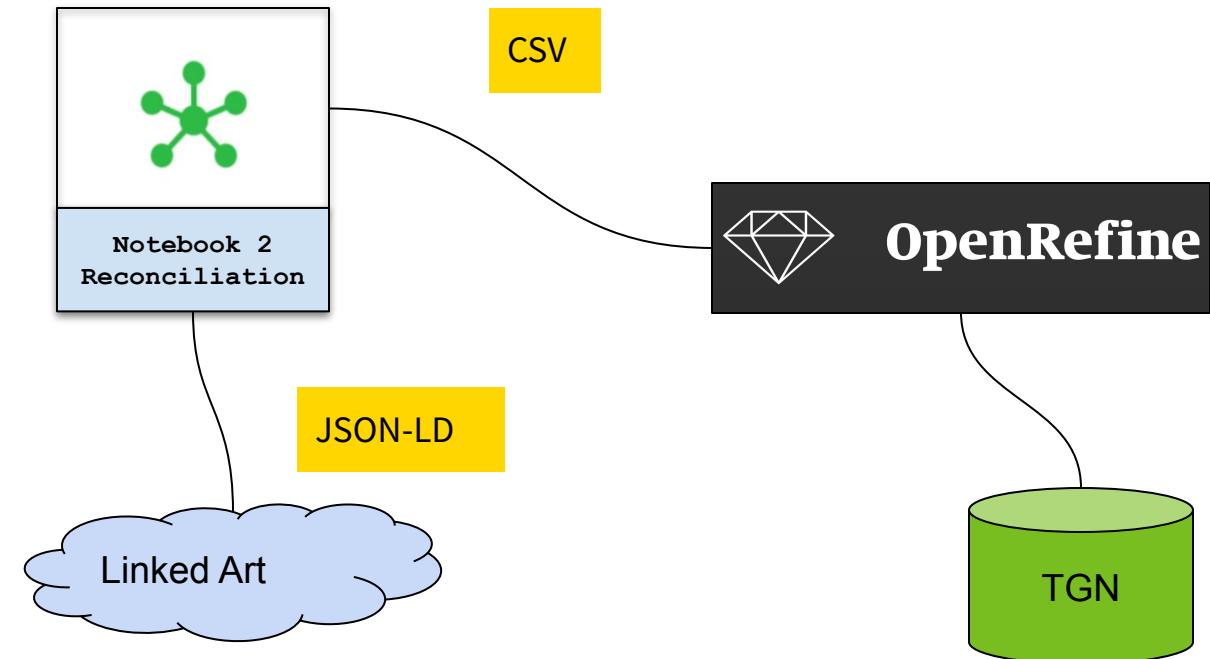
Reconciliation Code Notebook



- **Enrich data** with external data source
 - **Identify** place name
 - **Reconcile** place name with name authority
 - Getty Thesaurus of Geographical Names Online (**TGN**)
 - **Extract** authoritative global identifier for place name and geographical coordinates
 - **Add** new data into Linked Art data files



Reconciliation Code Notebook



Reconciliation Sub-steps: Identify → Reconcile → Extract → Add

Identify Place Name

Find place name in artwork title

Semi-manual process

Seed known locations

Result - CSV file with place name column

```
In [14]: artworkCsvFile = "./data/ruskin/ruskin-places.csv" # file location

# read CSV file into pandas dataframe
dataFrame = pd.read_csv(artworkCsvFile,low_memory=False)

# A list of place names `placeNames` is created to help with extracting place names from the artwork title.
placeNames = [
    "Florence", "Bologna", "Lucca", "Alps", "Oxford", "Rome", "Venice", "Fribourg", "Neuchâtel", "Sestri", "Visp", "Chamonix",
    "Abbeville", "Schaffhausen", "Verona", "Vorarlberg", "Baden", "Schaffhausen", "Faido", "Normandy", "Genève", "Geneva",
    "Gloucester", "Basel", "Lucern", "Padua", "Habsburg", "Rhine", "Zug", "Aix-la-Chapelle", "Siena", "Mont Blanc", "Lago di Como",
    "Bellinzona", "Lake of Lecco"
]

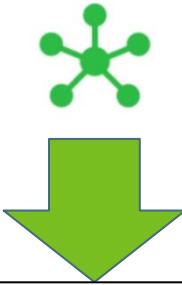
places = {"Venezia": ["Venice", "Venetian", "St Mark", "St. Mark"]}

# iterate over dataframe
for index, row in dataFrame.iterrows():

    # iterate over place names
    # check if any place name in placesNames is present in row
    for place in placeNames:
        # if place name found, add to place_modified column
        if place in row["place"]:
            dataFrame.at[index, "place_modified"] = place

    # iterate over place names for Venice
    for place in places["Venezia"]:
        # if place found add 'Venezia' to place_modified column
        if place in row["place"]:
            dataFrame.at[index, "place_modified"] = "Venezia"

# remove records where place_modified is blank
dataFrame = dataFrame[dataFrame.place_modified != ""]
dataFrame.to_csv(artworkCsvFile, index=False)
```



place	place_modified
Study of a Venetian Capital	Venezia
Autumnal Cloud filling the Valley of Geneva, t...	Geneva
Tom Tower, Christ Church, Oxford	Oxford
Study of a Venetian Capital	Venezia
View of Bologna	Bologna
...	...
Sketch of the Oak Spray in Mantegna's Fresco o...	Padua
The Garden of San Miniato near Florence	Florence
Part of a Sketch of the Northwest Porch of St....	Venezia
Gezicht op S. Anastasia te Verona, over de Adige	Verona
Study of the Marble Inlaying on the Front of t...	Venezia

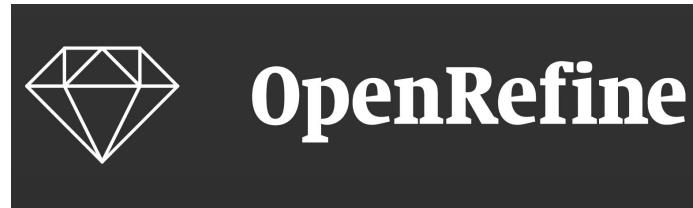
Reconcile

Uses

- CSV file with place name
- OpenRefine
- Getty Thesaurus of Geographic Names

Creates

- CSV file with column containing global identifier for place name



place place_modified tgn

Oxford	place_modified	tgn/7018159
	Facet	
	Text filter	
	Edit cells	
	Edit column	
	Transpose	
	Sort...	
	View	tgn/7018159
	Reconcile	Start reconciling...
	Alps	
	Lucca Sicula	
	Choose new match	
towards Lucca		
Vorarlberg		
Verona		
	Verona (29)	
	✓ Verona (29)	
	✓ Verona (29)	
	✓ Verona (29)	
	✓ Verona (29)	
	Create new item	
Schaffhausen		tgn/7106739
Baden		tgn/8707496

Add a column containing entity identifiers from items

place	place_modified	tgn
Venetian Capital	Venice	tgn/7018159
Church, Oxford	Oxfordshire	tgn/7011931
Venetian Capital	Venice	tgn/7018159
of Bologna	Bologna	tgn/7003127
of the Alps	Alps	tgn/7007746
...
Mark's, Venice	Venice	tgn/7018159
at Neuchâtel	Neuchâtel	tgn/7003751
Fresco o...	Padua	tgn/7003085
f the Northwest Porch of St....	Venice	tgn/7018159

Reconciliation Sub-steps: Identify → **Reconcile** → Extract → Add

Transformation → Reconciliation → Visualisation

OpenRefine



Create Project

New Version - Download OpenRefine v3.5.2 now.

[Create Project](#)

Start Over	Configure Parsing Options	Project name: rusin places.csv	Tags:
Create Project	Configure Parsing Options	rusin places.csv	Create Project
Open Project	Import Project	rusin places.csv	Verona
Language Settings	Parse Data	The North-West Angle of the Facade of St. Mark's, Venice	Verona
	Configure Headers	The Virgin della Peste, Verona	Verona
	Configure Text Processing	The Capodimonte at Lucern (Lucerne)	Lucern
	Configure Cell Processing	Mori Bianco from Saint-Martin-en-Ardo	Mont Blanc
	Configure Row Processing	Bergamo and the Alps, from the road to Bressana	Alps
	Configure Column Processing	The Palazzo Costabili-Passari, Verona	Verona
	Configure Row Groups	Pass of Faido	Faido
	Configure Cell Groups	Plaster on the unfinished Facade of Sant'Anastasia, Verona	Verona
	Configure Row Merging	The Gryphon bearing the south shaft of the west Entrance of the Duomo, Verona	Verona
	Configure Cell Merging	Sepia Sketch of Lautage, further named: Study from Ruskin's Photograph of the Courtyard of a late Gothic wooden House at Asolo	Asolo
	Configure Row Deletion	Design for a window in the University Museum, Oxford	Oxford
	Configure Cell Deletion	Convent and Alpine Pass, Map, Basement	Map
	Configure Row Insertion	Pass of Faido	Faido
	Configure Cell Insertion	Study in Colour of one of the Niches surrounding the Tomb of Ca' Signorile della Scala in Verona, with Remains of the 'Casa di Romeo'	Verona
	Configure Row Promotion	Design for a window in the University Museum, Oxford	Oxford
	Configure Cell Promotion	Studies in St. Mark's	Verona
	Configure Row Demotion	Lateral View of the Facade Said Website in Felt, Lucca	Lucca

[Parse data as](#) [Character encoding](#): [UTF-8](#) [Updates Preview](#)

[CSV / TSV / separator-based file](#) [Ignore first](#): [0 line\(s\) at beginning of file](#)

[New](#) [Parse next](#): [1 line\(s\) as column headers](#)

[Line-based text file](#) [Discard initial](#): [0 row\(s\) of data](#)

[Fixed-width field text files](#) [Load at most](#): [100000 row\(s\) of data](#)

[PCV/Acc field files](#) [Use character](#): [1 to enclose cells containing column separators](#)

[JSON file](#) [Parse cell text into numbers, dates,...](#)

[MAPC file](#) [Skip blank rows](#)

[OCV/LD file](#) [Skip blank cells as null](#)

[RCF/NC file](#) [Store file source \(file names, URLs\) in each row](#)

[Column names \(comma-separated\):](#)

Reconciliation Sub-steps: Identify → **Reconcile** → Extract → Add



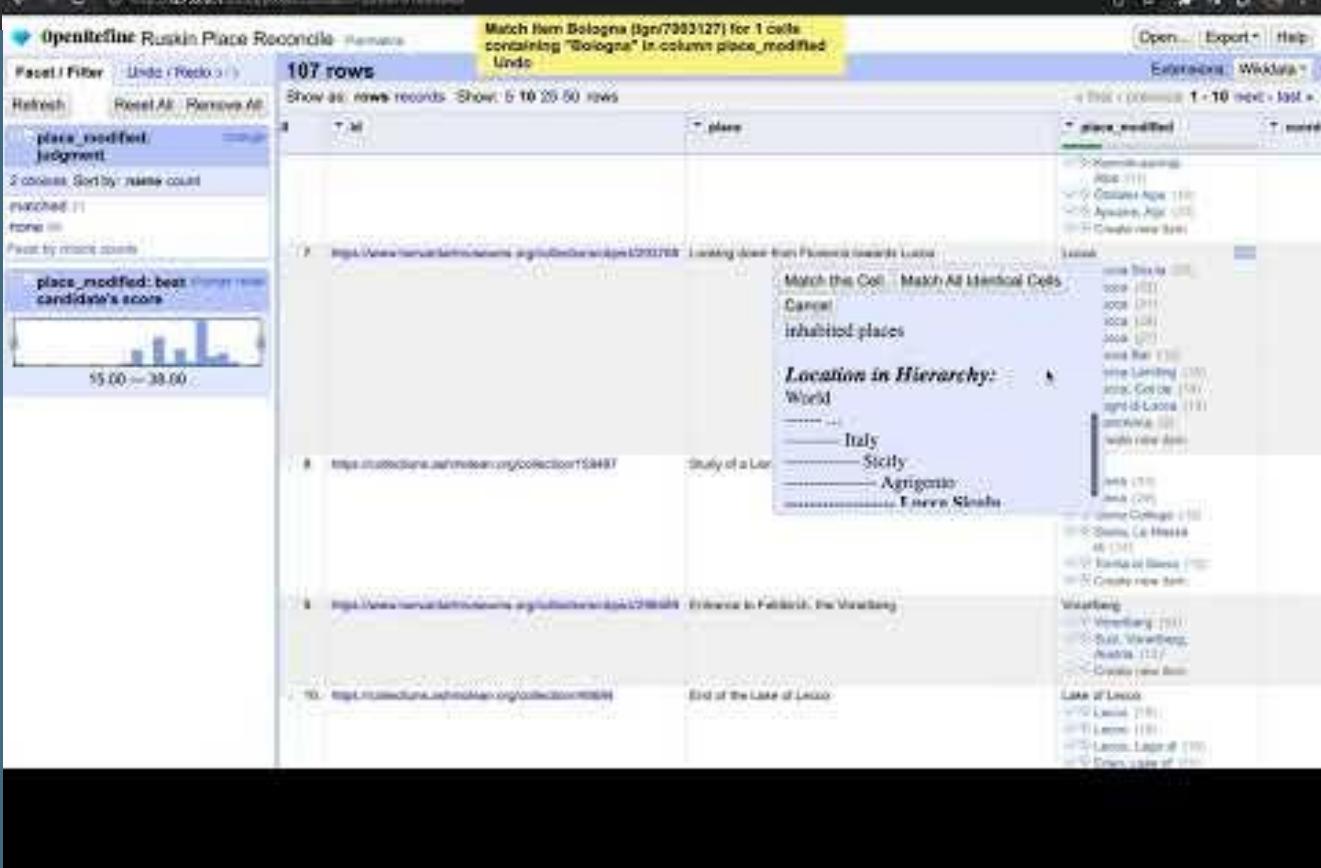
Reconcile

The screenshot shows the OpenRefine interface with the 'Reconcile' step selected. A modal dialog box is centered, displaying the message "Working..." with a circular progress indicator. The dialog also contains several configuration options:

- Reconcile column: "place_modified"
- Reconcile each cell to an entity of one of these types:
 - Wikidata item
 - Wikidata item (part-of)
 - Wikidata item (parent)
- Also use relevant objects from other columns
- Max entities per row: 100
- Maximum number of candidates to return: 100
- Reconcile against type: Wikidata item
- Reconcile ignore part-of type: Wikidata item (part-of)
- Auto-match candidates with high confidence: checked
- Add Standard Service...

Reconciliation Sub-steps: Identify → **Reconcile** → Extract → Add

Review Reconciliation Results

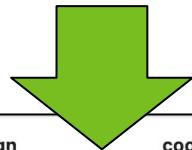


The screenshot shows the OpenRefine interface with the following details:

- Title:** OpenRefine Rustin Place Reconcile - [Parameter]
- Facet / Filter:** place_modified: judgment
- Panel:** place_modified: best candidate's score (histogram showing scores from 55.00 to 39.00)
- Table Headers:** ID, URL, place, place_modified
- Table Rows:**
 - ID 1: URL https://www.semanticscience.org/resource/12345678, place: Lucca, place_modified: Lucca
 - ID 2: URL https://www.semanticscience.org/resource/12345678, place: Lucca, place_modified: Lucca
 - ID 3: URL https://www.semanticscience.org/resource/12345678, place: Lucca, place_modified: Lucca
 - ID 4: URL https://www.semanticscience.org/resource/12345678, place: Lucca, place_modified: Lucca
 - ID 5: URL https://www.semanticscience.org/resource/12345678, place: Lucca, place_modified: Lucca
- Right Panel:** A detailed view of the first row (ID 1) with options to Match this Cell, Match All Identical Cells, Cancel, or Inhabituated places.
- Bottom Panel:** A tree view of the Location in Hierarchy: World, with branches for Italy, Sicily, Apulia, and Emilia-Romagna.
- Bottom Right:** A sidebar showing a list of locations and their counts, such as Lucca (10), Italy (7), Sicily (1), Apulia (1), Emilia-Romagna (1), and others.

Reconciliation Sub-steps: Identify → **Reconcile** → Extract → Add

Reconciliation: Extract



Extract additional data from name authority

- Geographical coordinates
- Python code
- Uses vocab.getty.edu TGN web service

```
for index, row in df.iterrows():
    gid = row["tgn"]
    if "tgn" in str(gid):
        infof = "http://vocab.getty.edu/tgn/" + gid.split("tgn/",1)[1] + "-place.json"
        response = requests.get(infof)
        json_data = response.json()
        for prop in json_data:
            lat= json_data[prop][ "http://www.w3.org/2003/01/geo/wgs84_pos#lat"][0][ "value"]
            lng = json_data[prop][ "http://www.w3.org/2003/01/geo/wgs84_pos#long"][0][ "value"]
            latlng = str(lat) + "," + str(lng)
```

place	place_modified	tgn	coords
tian Capital	Venice	tgn/7018159	45.438611,12.326667
rch, Oxford	Oxfordshire	tgn/7011931	51.75,-1.25
tian Capital	Venice	tgn/7018159	45.438611,12.326667
of Bologna	Bologna	tgn/7003127	44.466667,11.433333
		07746	46.416667,10
	
		18159	45.438611,12.326667
		03751	46.990867,6.797675
		03085	45.416667,11.883333
		18159	45.438611,12.326667
		18159	45.438611,12.326667

Get geocoordinates using TGN API



The screenshot shows a browser window with the following content:

```
# Create dataframe to hold geographical coordinates with columns tgn_id and tgn_tg
# DataFrameGeo = pd.DataFrame(columns=['tgn_id', 'tg'])

# Iterate through reconciled data, recovering place names and TGN identifiers
# for identifier_tgn in data['placeablePlaces']['tgn'].unique():

#     if identifier_tgn in data['placeablePlaces']['tgn'].unique():

#         # Create string for web service query of using .split()
#         query = "http://tgnweb.getty.edu/tgn?" + identifier_tgn + "&format=json&place=true"

#         # Use requests.get() to query TGN web service using TGN identifier to recieve geo-coordinates
#         resultsJSON = requests.get(query).json()

#         # get set of unique place service query results
#         for record in resultsJSON:
#             tgn_id = resultsJSON[record]['id']['tgn']
#             tg_id = resultsJSON[record]['id']['tg']

#             # Create string for lat/lng
#             latlng = str(tgn_id) + "," + str(tg_id)

#             # Append TGN identifier and lat/lng to the dataframe
#             DataFrameGeo = DataFrameGeo.append(
#                 {'tgn_id': tgn_id,
#                  'tg_id': tg_id,
#                  'latlng': latlng
#                 }, ignore_index=True)

#     # If there are no placeablePlaces with addition of new records
#     display(DataFrameGeo)
```

Geographical coordinates retrieved from TGN web service

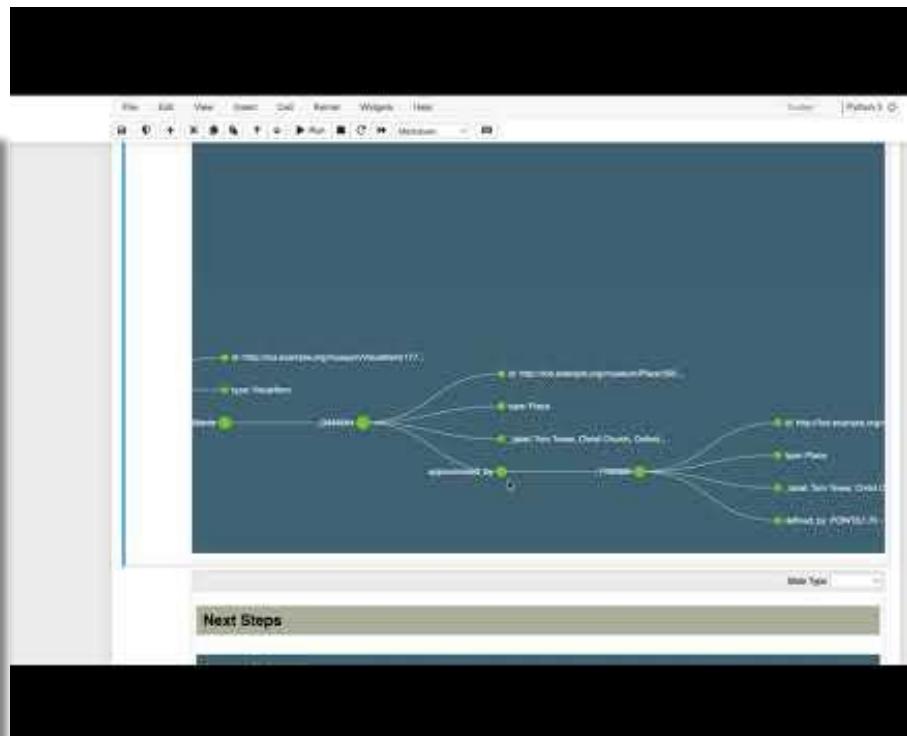
Retrieving geocoordinates from vocab.getty.edu TGN API. Please wait for task to complete.

Reconciliation Sub-steps: **Identify** → **Reconcile** → **Extract** → **Add**

Reconciliation: Add Geo Coords

<https://linked.art/model/>

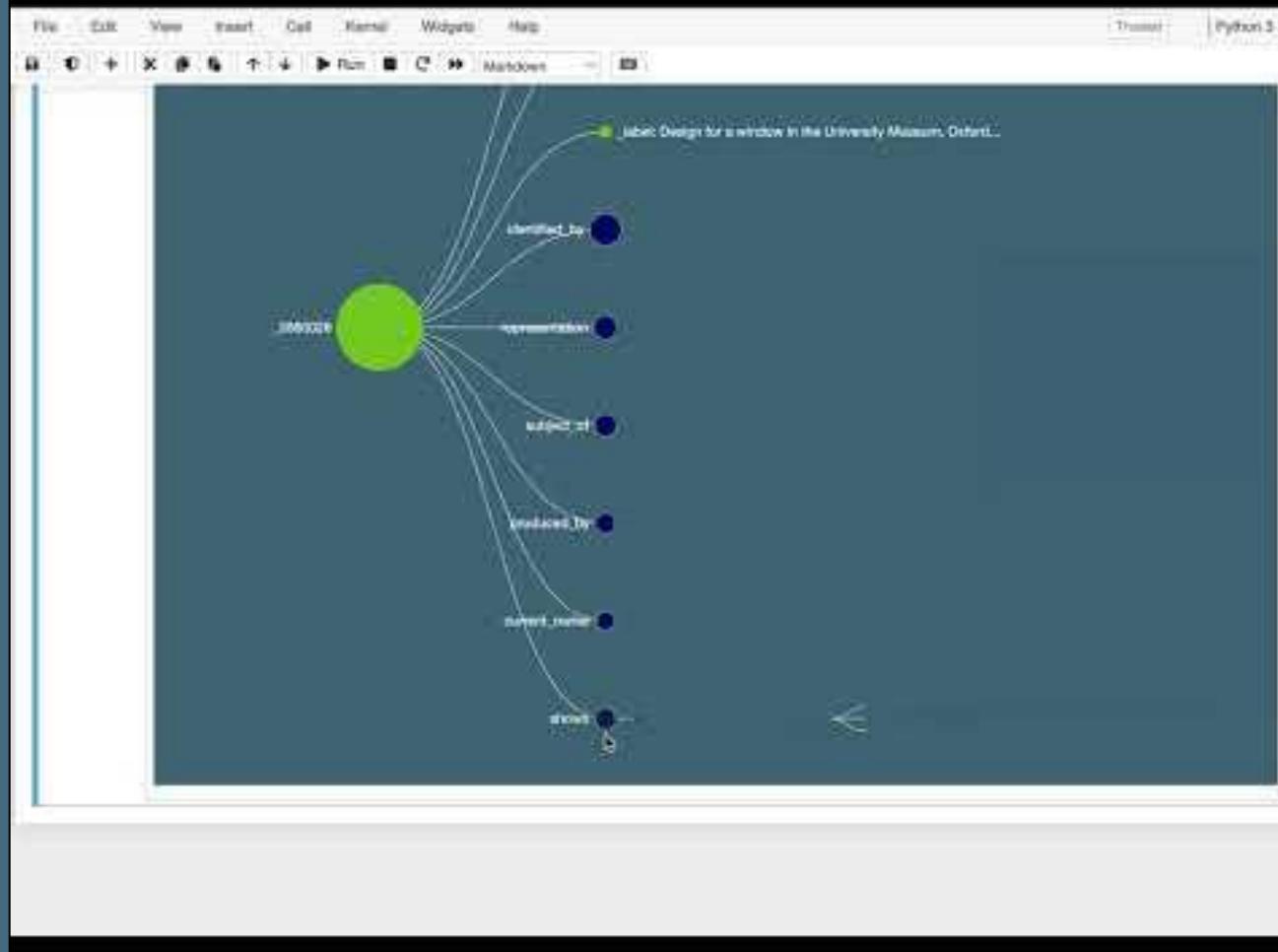
```
{  
  "@context": "https://linked.art/ns/v1/linked-art.json",  
  "id": "https://linked.art/example/object/34",  
  "type": "HumanMadeObject",  
  "_label": "geographical place name",  
  "shows": [  
    {  
      "type": "VisualItem",  
      "represents": [  
        {  
          "type": "Place",  
          "_label": "Lucca",  
          "approximated_by": [  
            {  
              "type": "Place",  
              "_label": "Lucca - Location Approximation",  
              "defined_by": "POINT(-0.0032937526703165 51.515107154846)"  
            }  
          ]  
        ]  
      ]  
    }  
  ]  
}
```

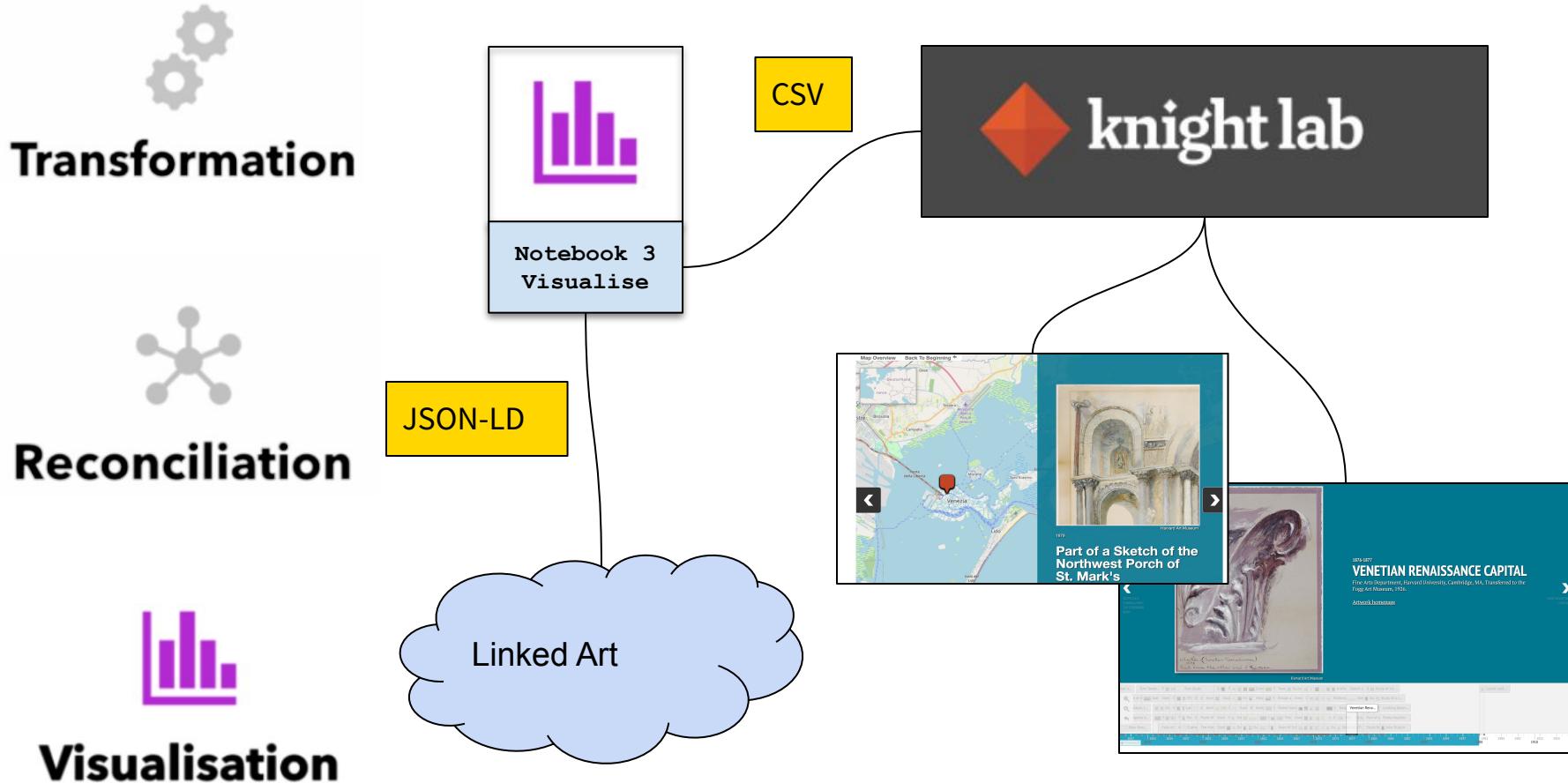


John Ruskin

Reconciliation Notebook

Visualisation of
JSON-LD for selected
artwork





Visualisation

Uses

- KnightLab visualisation
- Linked Art JSON-LD
- Script to transform to CSV/JSON

Creates

- Timeline
- storymap

The screenshot displays two web-based visualization tools developed by Knight Lab. On the left, the **StoryMap JS** interface is shown. It features a map of Eastern Europe with a red marker indicating a location in Donets'k. A dashed line connects this location to another red marker in Vohovakha. The map also shows several other locations with grey markers: Kurakhovo, sinovataya, Nizhnyaya Krynka, Khartsyzsk, Ilovays'k, Mospyne, Norvy Svit, Starobesheve, and Dokuchayevsk. An inset map shows the location of the main map within Europe. Below the map, the text "StoryMap JS" is displayed, followed by "Maps that tell stories." On the right, the **Timeline JS** interface is shown. It consists of a horizontal timeline with vertical markers for specific dates. Events listed include: "John Kerry visits..." (Jan. 28), "Concerns over..." (Feb. 1), "Launch of 'missile'" (Feb. 4), "South Korean workers leave Kaesong industrial park" (Feb. 10), and "North Korea..." (Feb. 22). The timeline continues into March. Below the timeline, the text "Timeline JS" is displayed, followed by "Easy-to-make, beautiful timelines."

Timeline Notebook

A screenshot of a Jupyter Notebook interface. The main content area displays a black and white portrait of John Ruskin, a man with dark hair and a high-collared coat. Below the portrait, the text "JOHN RUSKIN" is displayed in large, bold, black capital letters. Underneath the name, a subtitle reads: "This timeline visualization shows artworks created by John Ruskin." A descriptive paragraph follows: "It demonstrates how the Linked Art data model can be used to search for, record and visualize collective data for artworks." At the bottom of this section, there is a link "See linkedart for more information." To the right of the main content, there is a vertical sidebar with some icons and text.

StoryMap Notebook

The screenshot shows a StoryMap Notebook interface. On the left is a map of Europe with several dark grey polygonal shapes representing travel routes. A specific route is highlighted with a thick black border. To the right of the map is a detailed sidebar for "John Ruskin's Travels in Europe".

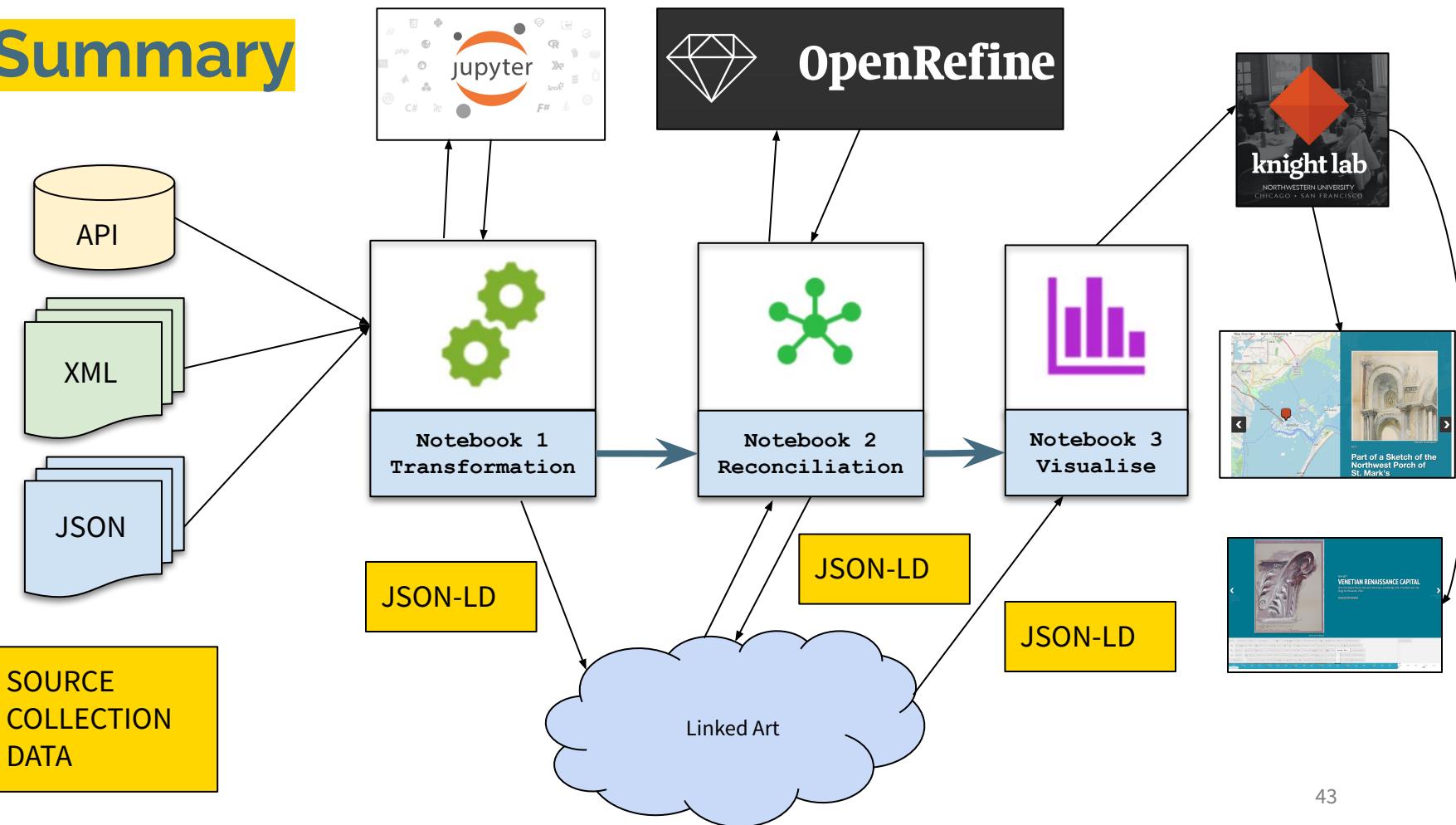
John Ruskin's Travels in Europe

John Ruskin

From Wikipedia, the free encyclopedia

John Ruskin, 3 February 1819 – 20 January 1900, was an English writer, philosopher, and one of the principal figures in the Arts and Crafts movement. He wrote on subjects as varied as geology, architecture, myth, ornithology, literature, education theory, and social issues.

Summary



Acknowledgements



Arts and
Humanities
Research Council

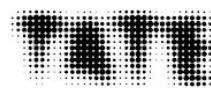
This work was undertaken by the [Linked Art II project](#) at the University of Oxford (Principal Investigator: Dr. Kevin Page, Oxford e-Research Centre) funded by the [UK Arts and Humanities Research Council](#) (AHRC project reference AH/T013117/1). The project's Research Software Engineer was Tanya Gray.

We gratefully acknowledge the participation and contributions of our **project partners** and the wider **Linked Art community**.



The work was supported by the Centre for Digital Scholarship @ Oxford (DiSc).

Thank you to the museums and galleries that are making their collection data available for re-use, via APIs and data downloads



Questions & Answers

Next Steps

- **Explore**
 - (and modify) the code notebooks
- **Complete**
 - the Linked Art Questionnaire that seeks feedback on the notebooks, Linked Art and invites collaboration
- **Register**



for the Linked Data strand of the
Digital Humanities @ Oxford
Summer School (DHOxSS)

Code Notebooks

- <https://github.com/tgra/Linked-Art/>

Linked Art Questionnaire

- <https://linked.art/questionnaire/>

Linked Art Data Model & Community

- <https://linked.art>

Digital Humanities @ Oxford Summer School (DHOxSS)

- <https://dhoxss.net>