Baroque AI: Publication Prototype

Class participants

3/17/23

Table of contents

Chapter 1

Part of the series: Baroque TOC

Programme instructions

2023-03-17 v1.0

Venus und Cupido, Heinrich Bollandt, between circa 1620 and circa 1630. https://commons.wikimedia.org/wiki/File:Heinrich_Bollandt_-_Venus_und_Cupido.jpg This work is in the public domain.

Example publications:

- Exhibition catalogue demo: toc Baroque /toc from Experimental Books – Re-imagining Scholarly Publishing, COPIM. Workshop URL: https://experimentalbooks.pubpub.org/programme-overview
- Publishers catalogue demo: ScholarLed A catalogue of ScholarLed presses built on a Quarto / Jupyter Notebook model for computational publishing. The publication is automatically updated daily to reflect any new books added by the publishers.
- Proof of concept #1 - Computational Publication: Computational Publishing for Collections - ADA CP Prototype #1 - Nov 22
- Proof of concept #2 - To be confirmed, completion for end of April 2023. This contains all parts fully rendered: Cover, colophon, essay, collection, graph, TIB AV Portal, Semantic Kompakkt
- semanticClimate: To be confirmed - customised research papers readers made for regional climate change action plans based on IPCC reports and sourcing content from open research repositories.

- FSCI Summer School - publishing from collections class: To be confirmed, July 2023

Chapter 2

Colophon

PUBLISHING FROM COLLECTIONS USES OF COMPUTATIONAL PUBLISHIGN AND LINKEDOPEN DATA

Open Science Lab - TIB Hannnover

First published 2023-03-30

DOI: https://doi.org/10.5281/zenodo.7701161

Chapter 3

Catalogue Experiment: Baroque AI

`Nextcloud Markdown document link: https://tib.eu/cloud/s/qBx8SbqiPBBedye`

3.1 Part of the series: Baroque TOC

- Class instructions and all links: https://nfdi4culture.github.io/class-ADA-CP-pipeline/
- Demo publication: https://nfdi4culture.github.io/catalogue-003/
- Repo link: https://github.com/NFDI4Culture/catalogue-003

3.2 Add your name:

- Simon Worthington

3.3 Text editing

Paste in a section of text based on variation of Baroque painting collections in the state of Bavaria.

https://openai.com/blog/chatgpt

https://www.perplexity.ai/

Chapter 4

Activity: Paintings catalogue in Jupyter Notebook

Objective: Make a selection of nine paintings for the exhibition catalogue to be selected from Wikidata and rendered multi-format in Quarto.

The below Python code uses SPARQLWrapper to retrieve data from Wikidata based on a SPARQL query.

Wikidata link: http://www.wikidata.org/entity/Q25569472

Title: Judith

Year: 1625

Creator: Simon Vouet

Copyright: public domain

Wikidata link: http://www.wikidata.org/entity/Q27316110

Title: Zeil- en roeiboten in een riviermonding

Year: 1640

Creator: Jan van Goyen

Copyright: public domain

Wikidata link: http://www.wikidata.org/entity/Q27919851

Title: The Gallery of Archduke Leopold in Brussels (I)

Year: 1651

Creator: David Teniers the Younger

Wikidata link: http://www.wikidata.org/entity/Q27976019

Title: A Woman Playing a Lute

Year: 1663

Creator: Frans van Mieris the Elder

Copyright: public domain

Wikidata link: http://www.wikidata.org/entity/Q27976019

Title: A Woman Playing a Lute

Year: 1663

Creator: Frans van Mieris the Elder

Copyright: public domain

Wikidata link: http://www.wikidata.org/entity/Q28379861

Title: Portrait of David II Teniers

Year: 1659

Creator: Pieter Thijs

Copyright: public domain

Wikidata link: http://www.wikidata.org/entity/Q28795868

Title: The Banishment of Hagar

Year: 1668

Creator: Claude Lorrain

Copyright: public domain

Wikidata link: http://www.wikidata.org/entity/Q28795872

Title: Q28795872

Year: 1665

Creator: Francisque Millet

Copyright: public domain

Wikidata link: http://www.wikidata.org/entity/Q28801542

Title: Madonna and Child

Year: 1635

Creator: Philippe de Champaigne

Chapter 5

Activity: Embedded video in Jupyter Notebook

Objective: Running and editing Juypter Notebooks in MyBinder and retrieving video and 3D models as embeds.

The below Python code experiments with retrieving video data via iframe embedding.

```
<IPython.core.display.HTML object>
```

5.1 3D model embedding

The below Python code experiments with retrieving 3D data via iframe embedding.

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```