Closing the Gap in Non-Latin Script Data

Pragmatic methods to manage corpora with diverse languages

Aim of the project

- provide an overview over projects that work with non-latin script
- provide insight into
  - tech stack
  - tools
  - policies
  - contacts
- provide a best practice example for corpus management

Basic needs

- no conventional setup required (no server-side infrastructure)
- should survive longer than the usual third-party funding period
- no "could I please have"-infrastructure
- use established workflows and industry standards
- **FAIR**
- transparent
- easy and nice frontend (initial phase: **Vue.js**, now: **SvelteKit**)
- as static as possible
- clear separation between data and frontend
- accessible from (almost) everywhere

Initial ideas

- established formats: `.json`, `.xml` or `.yaml`
  - `.json` as the first choice for its widespread use and comparably easy use with non-latin Script
- **GitHub** for repository and public issue management
- **GitHub Pages** for frontend
- implementing of taxonomies and existing standards
- implementing **CRediT**
- CC-BY license
- heavily committing to **Open Science**
- every major decision should be documented transparently
- "practice what we preach": provide an example to show it is actually possible to work **FAIR**.

Why not GitHub?

- owned by **Microsoft**, thus we use "Big Tech" solutions instead of local academic infrastructure
- data is "out in the wild"
  - this is an issue for sensitive data

Why GitHub?

- data is "out in the wild"
- **GitHub** has a higher probability to be working even in 5 or 10 years
- accessibility without academic credentials
- **GitHub Actions** and **GitHub Pages**
- future alternatives can be non-profit providers, e.g. **Codeberg**
- project and issue management resources to allow collaboration

Real Talk

I would prefer academic infrastructure, but there is often no way or will to provide for a platform that is as accessible, easy to use and long-term supported as providers like GitHub. It is just not possible yet to realize a similar setup compared to ours in a conventional academic context (and in an appropriate timespan).

Closing the Gap Database

folder structure:

```
/PROJECTS/
.. {simplified_project_name}/
..  .. {uuid}.json
/DOCS/
/KEYWORDS/
/SCHEMATA/
...
/projects.json
```

the `projects.json` contains a mapping of `{uuid}` to `{simplified_project_name}`, resulting in a:

- human-readable folder structure
- machine-readable UUID-mapping

Closing the Gap Datamodel

- versioned and **Zod**-validated `json` schema
- multiple sections:
  - metadata on the `.json`-file
  - metadata on the project
  - metadata on the relations of the project
- example: **AnonymClassic** (`d1e6d69b-5e9a-4b4a-85ad-09aac56ed2d9`)

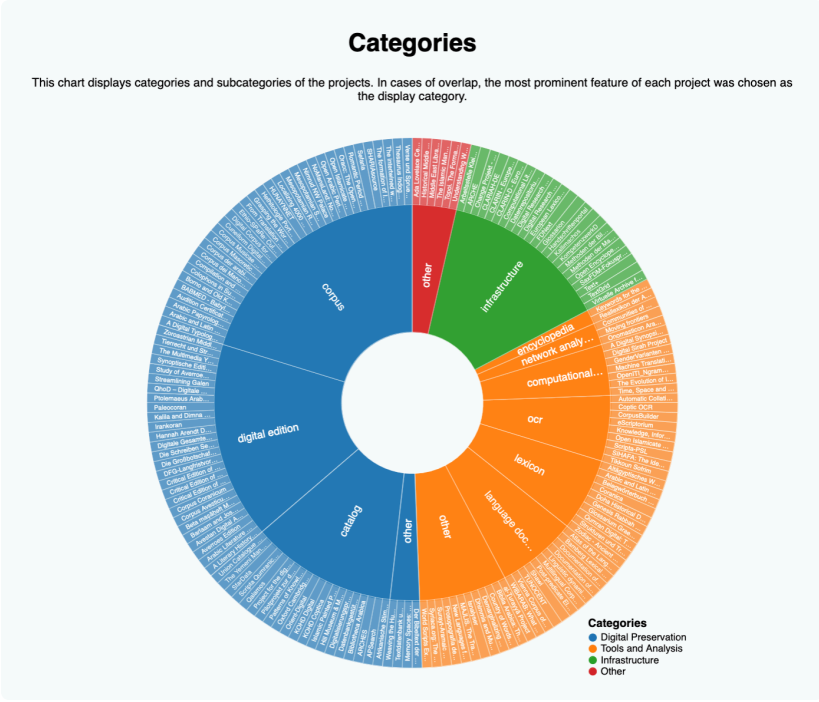

```
{
  "schema_version": "0.2.4",
  "record_metadata": {
    "uuid": "d1e6d69b-5e9a-4b4a-85ad-09aac56ed2d9",
    "record_created_on": "2021-11-08",
    "record_created_by": "Kudela, Xenia Monika",
    "last_edited_on": "2022-02-18",
    "interviewed": false
  },
  "project": {
    "title": "Kalila and Dimna - AnonymClassic",
```

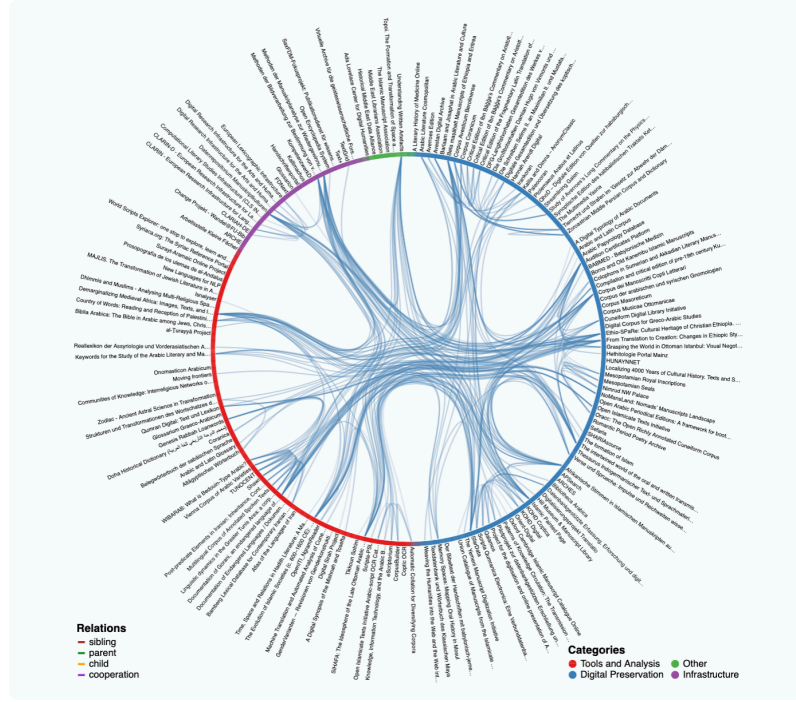
Additional workflows:

- regular releases on **GitHub** and **Zenodo** of the complete project (DOI: `https://zenodo.org/doi/10.5281/zenodo.8329145`)
- monthly snapshots of the frontend in the **Web Archive** (`https://web.archive.org/web/20240000000000*/https://m-l-d-h.github.io/Closing-The-Gap-In-Non-Latin-Script-Data/`)
- regular link-checks via **Lychee**

Additional services

- link-checking helps to track dying project websites or unstable links
- support in networking and flat corpus management
- data analysis for academic politics



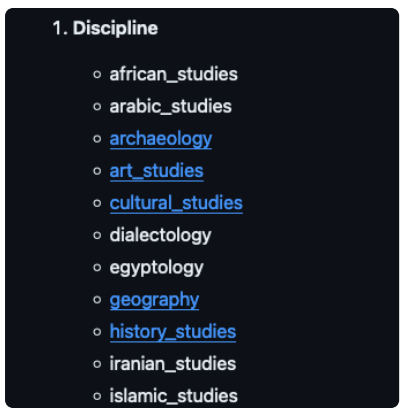


Issue 1: Taxonomies

- implementation of taxonomies for descriptive keywords (e.g. **DHA**)
- **ISO 639-3** codes for languages

BUT: non-latin script related concepts may not be supported by existing taxonomies (e.g. "arabic_studies").

workaround: mapping a *short* self-developed taxonomy to established taxonomies.



Issue 2: Data acquisition

How to get comprehensive data?

as so often: the project requires people to provide data, but people are not always willing to do so:

> "we have better things to do"
> "this is OUR data"
> "someone may steal the data"
> "I'd rather use google"

and soon probably:

> "what about chatGPT and artificial intelligence?"

Conclusion

The CtG project provides a metadata corpus on projects working with non-latin scripts

The infrastructure…

- requires no conventional setup and sysadmin
- is free of charge and accessible from (almost) everywhere
- provides multiple methods to provide sustainable long-term archiving (github, web archive, zenodo)
- provides human- and machine-readable data for further processing
- provides insight in topics that are commonly ignored (e.g. the wide lack of sustainability plans)
- provides an overview over different stacks, tools and methods in use
- is independent from limited third party funding

… but …

- may be problematic with sensitive or complex data
- requires services like github actions and github pages to stay alive to be fully functional
- requires a certain amount of tech affinity to allow for collaboration without further assistance

Last Slide

Thank you for your attention. Questions?

backend

frontend