GN Meeting 2020
A better GeoNetwork user experience
1. Introduction

What’s up?

- GeoNetwork 4 around the corner!
- One huge change in the architecture (Lucene ➡ Elastic Search)
- Several impacts on the current UI, but…
  ...did not warrant a larger refactoring.

- Thus, GN4 will *feel familiar* to users of the previous version.
1. Introduction
1. **Introduction**

*Present-day struggles...*

- **Showcase:** **Sextant**, a catalog managed by the Ifremer
  - **Embeddable** catalog on third party websites (with many options)
  - **Custom themes** for each
1. Introduction

...And emerging opportunities

- New interesting use cases have appeared over the years
  - Making several UI themes for different organizations
    - Addressed in GN3.8 with portals
  - Including only parts of a catalog in a host page, e.g. search results, full record…
  - Integrate better with other frontend applications (map viewer, metadata editor…)

- Recent JS frameworks are getting faster and lighter
- Browsers are stronger than ever!
  - Web Components to replace <iframe> and the like
  - CSS variables to allow dynamic styling
1. Introduction
1. Introduction

What’s keeping us then?

● The current UI is **embedded** in the Java application
● This is a longstanding situation that comes with limitations:
  ○ Almost no way to **migrate away from AngularJS** (EOL is June 2021)
  ○ Complex and non-standard build system
  ○ Awkward to test and debug
  ○ A custom UI will require a **complete fork** of GeoNetwork
  ○ No way to **keep up with the evolutions** of the JS ecosystem!
● ...and with merits:
  ○ **Easy to deploy**: install once, get everything!
  ○ No cross-domain troubles, facilitate backend-frontend communication..
1. Introduction

*But I like the UI!*

- We do too! It is:
  - Extremely feature rich and mature
  - The result of *years of collaborative work* (thanks!)
  - Highly customizable with an extensive list of settings

- Unfortunately, it is also *very costly* to work with and maintain
1. Introduction

- Let’s face it:

  The GeoNetwork UI has reached a state where it cannot be refactored significantly anymore.

  …and that’s totally fine!

https://www.deviantart.com/built4ever/art/Castle-and-Village-Number-Two-334499398
2. A new frontend for GeoNetwork
2. A new frontend for GeoNetwork

*Some forward thinking*

- Overcoming the current limitations means *rewriting the UI* from scratch
- This *does not have to be destructive*: both the “present” and the “future” UI can live side-by-side, fulfilling different needs
- **Maintenance** efforts could go to the present UI, while *new features* could be directed towards the future UI
  - Less new features on the present UI means *less maintenance required* as well!
2. A new frontend for GeoNetwork

What benefits? (1)

- A complete rewrite would allow using a more recent framework, giving...
  - Better performance
  - Faster page loading
  - Better accessibility
  - Better code quality meaning...
  - Lower maintenance cost
  - More open to contributions
2. A new frontend for GeoNetwork

*What benefits? (2)*

- Revising the UI architecture might also give:
  - Better separation of concerns
  - No more *monolithic* structure, easier to migrate/evolve (= future proof)
  - Lower development costs
  - No full *vendor lock-in*
  - *Pre-rendered* content (better SEO!)
  - *Lazy load* parts of the application
  - Better *developer experience!*
A different approach

The so-called “GeoNetwork UI” could offer much more than nowadays

Instead of being a “one-size-fits-all” app, it could provide...

- Several smart components usable in larger apps
- Embeddable “mini-apps”, portable and working whatever the context
- A couple of full-blown apps similar to the existing UI, using said components
2. A new frontend for GeoNetwork

_Smart components... smart how?_

- Modern JS development often use the concept of smart/dumb components
- Smart components are responsible for _fetching_ and _preparing_ the data
- They typically interact with each other using a "store"

- In a larger app with a _similar framework_, these could blend in seamlessly!
2. A new frontend for GeoNetwork

*Smart components… smart how? (2)*

- In a larger app with a similar framework, these could *blend in* seamlessly!
2. A new frontend for GeoNetwork

Smart components… smart how? (2)

- Practical example: an app providing a single entrypoint to data & metadata
  - The app could show a **single text input** which allows the user to search both for data services and metadata records
  - The text input and data-related results would be provided by the **app components**
  - The metadata records would be provided by the **GeoNetwork components**
  - **Spatial search** could also be implemented

---

**App components**

- 32 data objects found.

**GeoNetwork components**

- **Schools in the region**
  - Last updated: two weeks ago
  - Status: continuously updated
  - Provider: Académie

- **Access to education by territory survey**
  - Last updated: one year ago
  - Status: complete
  - Provider: Inspection

---

**Overview of secondary schools in the region**

Last updated: two weeks ago

Status: continuously updated

Provider: Académie

**Feature**

Type: adm:schools
Attributes:
- name: Lycée de Monfermeil
- students: 83
- personnel: 14

**Feature**

Type: adm:schools
Attributes:
- name: Lycée de Montargis
- students: 134
- personnel: 26

...
2. A new frontend for GeoNetwork

Smart components… smart how? (3)

● **Summary:**
  ○ Basic **building blocks** of any future GeoNetwork UI
  ○ Efforts put into these will **benefit to all consumers** down the line
  ○ **Loosely coupled** and composable (components do not depend on each other)
  ○ Useable in any other app using the same framework!
2. A new frontend for GeoNetwork

*Embeddable “mini-apps” (1)*

- Very **easy** to include:
  - Only **one file** to load
  - Add an element in the HTML with the required input & style, e.g.
    ```html
    <gn::quicksearch
      api-url='https://mycatalog.org/geonetwork/srv/api'
      org-filter='sample-organization'
      main-color='#FF328C'
      secondary-color='lightgrey' />
    ```
  - No **requirements**, no **conflicts** with the host page
  - Uses the **WebComponents** standards (W3C), widely adopted
- Uses GeoNetwork **smart components** internally
- Each mini-app is **tailored** for a specific use case
2. A new frontend for GeoNetwork

Embeddable “mini-apps” (2)

- Some examples:
  - Providing a simple search interface (one text input, some results)
  - Showing a full metadata record
  - Basic map viewer showing either metadata extent or WMS/WFS data
2. A new frontend for GeoNetwork

*Embeddable “mini-apps” (3)*

- Summary:
  - Very simple to use, accessible to non-developers
  - Opens up new ways to access the metadata catalog in various contexts
  - Can blend in a surrounding theme with dynamic styling
  - Will require extra maintenance for those specifically
2. A new frontend for GeoNetwork

*Full-blown apps (1)*

- A small number of **feature-complete** apps can be built using smart components.
- Could be similar to the current UI logic, although…
- Offering more **specialized** apps might make sense, e.g:
  - Metadata search *without* map viewer
  - Metadata search *with* map viewer
  - Metadata authoring tool
- Specialized means **lighter** and **better suited** for their intended usage.
2. A new frontend for GeoNetwork

*Full-blown apps (2)*

- Such apps could be deployed as *stand-alone projects* (e.g. docker image)
- Also possible to *embed* them in the Java webapp to allow hassle-free deployment (same as the current system)
2. A new frontend for GeoNetwork

Full-blown apps (3)

- **Summary:**
  - Similar experience to the existing GeoNetwork UI
  - Could be deployed individually or embedded in the Java webapp
  - Addresses broader use cases, gives access to the catalog in a more traditional way
3. Starting point
3. Starting point

June 1st/2nd Codesprint (1)
3. Starting point

*June 1st/2nd Codesprint (1)*

- This Codesprint was focused on laying down the foundations for a new GeoNetwork frontend

- **Goals were:**
  - Make motivated choices for the technological components
  - Set up a skeleton of the project structure
  - Build a basic POC showing some of the expected benefits

3. Starting point

*June 1st/2nd Codesprint (2)*

- The project can be found at [https://github.com/geonetwork/geonetwork-ui](https://github.com/geonetwork/geonetwork-ui)
- Contains:
  - A few smart components talking to each other: search text input, search results, sort by button…
  - A stand alone app using these components
  - An embeddable web component using these as well
  - Automated code checking using Github Actions
  - A “story book” to review individual presentational components
3. Starting point

*June 1st/2nd Codesprint (3)*

- A few **priorities** emerged:
  - Separation of concerns: components dedicated either to presentation or logic
  - Low compiled file size (= faster loading)
  - Straightforward build system
  - Dynamic theming
  - Ability to produce web components
  - Code quality: type-based language, code formatting, automated tests, guidelines
3. Starting point

What next? (1)

- **GN community** (you!) will drive the priorities and help establish the road map
- Possible topics:
  - Search components (facets, sorting, advanced search…)
  - Metadata record formatter
  - Permalink management
  - Data downloading
  - Map viewer
  - Pre-rendered pages / SEO
  - Other?
3. Starting point

What next? (2)

- Embeddable mini-apps can be released along the way, as more smart components become available
- Keep maintaining the existing GN4 UI, while avoiding investing in it too much
- Admin UI modules will not benefit from a complete rewrite as much; should be lower priority
- Depending on backend architecture changes, separate UI apps could be made for catalog administration, editing, harvesting records...
3. Starting point

...and what about metadata editing?

- The current editor is frustrating for a lot of people
- Metadata editing is a complex topic that spans both backend and frontend concerns
  → Out of scope for this presentation
- The new UI architecture would open up new options for a separate editor application, which...
  - Could use the future UI components to list records, subtemplates, services...
  - Could be integrated as a component as well, or inside a dedicated “mini-app”
4. Conclusion
4. Conclusion

- The current GeoNetwork UI is staying:
  - One-stop-shop mindset, easy install, customizable!
  - No additional development cost!

- New development efforts go to a separate UI project:
  - New features will not break, i.e. no regressions!
  - Future-proof architecture
  - Many more options to access the catalog and its features
  - Each € spent will be more profitable!
4. Conclusion

Any questions?