Data Access Automation at Scale

Miró Khalifa
Data Access is a fundamental privacy expectation
Query user
Query posts
Create an HTML template
Format the content
Package
Compress
Data relations can be complex; even preparing seemingly simple data types can be a time-consuming endeavor.
New data types require repeated efforts numerous times.
Moreover, updates to data types should keep privacy expectations up-to-date.
Data is stored in various systems and databases.

Data is supported by hundreds of engineering teams.
$annotations

->userDataAccess()

Engineers are provided with an annotations API, and code is generated to:

1) **Safely retrieve** the data
2) **Format** it in unified containers
3) **Expose** it in rich formats
Schematization Framework

Annotations API

Data Safety

Privacy Safety

Unified Querying

Asynchronous Tier

Data Access Infrastructure
01 A reliable infrastructure
A reliable infrastructure, scales to user data needs, including:

The **amount of data** being served, and the **number of requests** to serve that data.
02  Data annotations
Data annotations to automate the process
Data actors

Data actors identify the subject of a data point.

Policy scopes

Policy scopes state the reasons for which our policy does or does not apply.
Fetchers

*Fetchers* specify how to load the data, e.g. from an edge or userId field

Fields

*Fields* specify the policies applied on fields

Edges

*Edges* allow hopping to connected edges to gain expanded or contextual data
Data Fetching

- `fetchers`
  - `Fetcher("")`
    - `category(..)`
    - `scope(..)`
    - `filter(..)`

- `fetchers/fields`
  - `Fetcher/Field("")`
    - `title(..)`
    - `description(..)`

- `fields/edges`
  - `Field("")`
    - `header/body()`
    - `inline/block()`
    - `primary/Timestamp()`

Content

Design
The schema language includes a validation step, which contributes to the semantic correctness of the annotations, specifically:

(a) The ownership relations of the data,
(b) how to load and render it, and
(c) how to explain it in a meaningful way.
$annotations
   ->userDataAccess()
   ->fetchers(...)

fields(): {
   Field(''),
   ->userDataAccess(...),
   Field(''),
   ->userDataAccess(...),
}

edges(): {
   Edge(''),
   ->userDataAccess(...),
   Edge(''),
   ->userDataAccess(...),
}

generated code:

fetchData(...)
getAccessors($fetcher, ..);
$context = appendContexts($edges, ..)
$unifiedQueryingLanguage();
$privacySafeQuery();
A UI can visualize and annotate schemas within a few clicks
03 Flexibility and enforcement
Enforcement of this process can occur during schema creation

A \texttt{\textasciitilde developmentOnly()} annotation can facilitate prototyping before a product/feature launch
Takeaways

01 Data access at scale requires a reliable infrastructure

02 Data annotations are efficient for automating and enforcing data access
Thank you.

MiroKal@meta.com

Images credit: DALL-E