Developing Web Applications for DHIS2

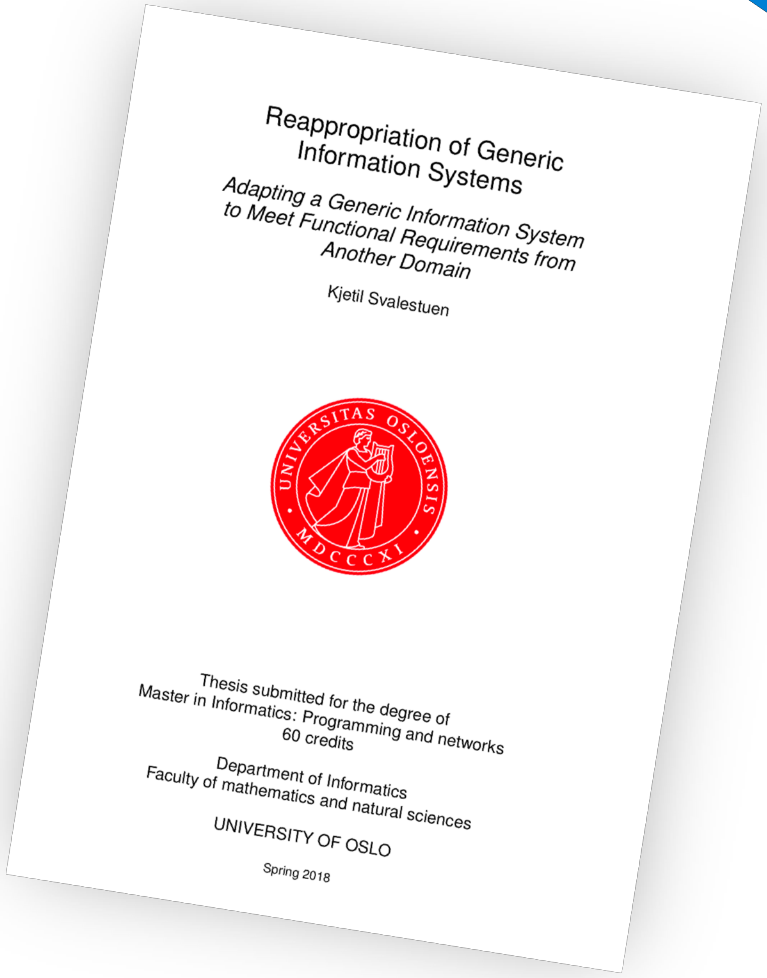
An introduction to the web API and how to create applications for it

IN5320 – Development in Platform Ecosystems

Kjetil Svalestuen

About me

- Works as a consultant at Bekk
  - Currently stationed at NAV
- Previously student at IFI
  - Took this course in 2016
  - Master thesis on DHIS2
    - *Reappropriation of Generic Information Systems*
- Previously part-time developer at DHIS2
  - Worked mainly on the *Maintenance* and *Scheduler* apps



Reappropriation of Generic Information Systems

Adapting a Generic Information System to Meet Functional Requirements from Another Domain

Kjetil Svalestuen

Thesis submitted for the degree of Master in Informatics: Programming and networks
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

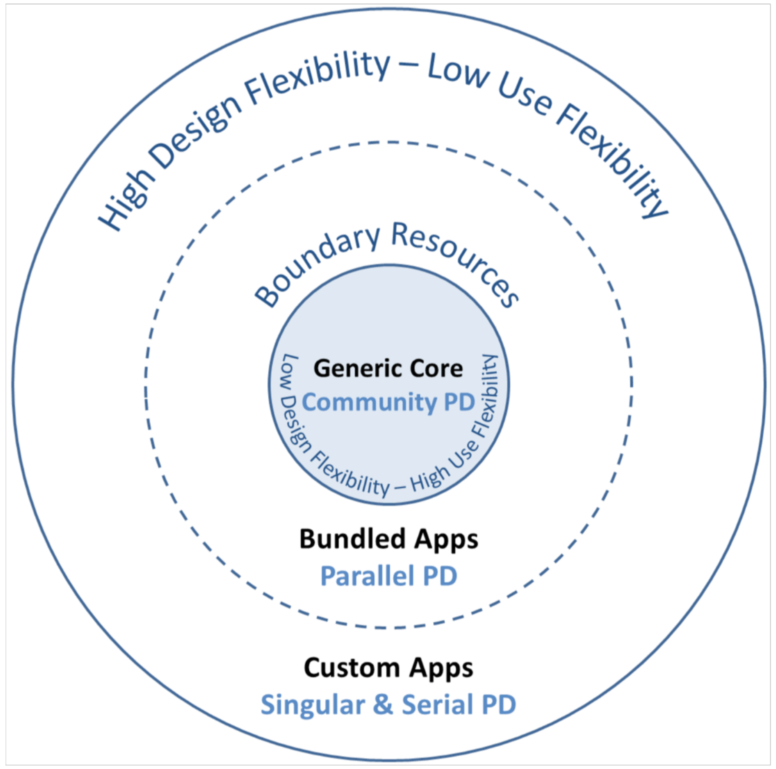
UNIVERSITY OF OSLO

Spring 2018

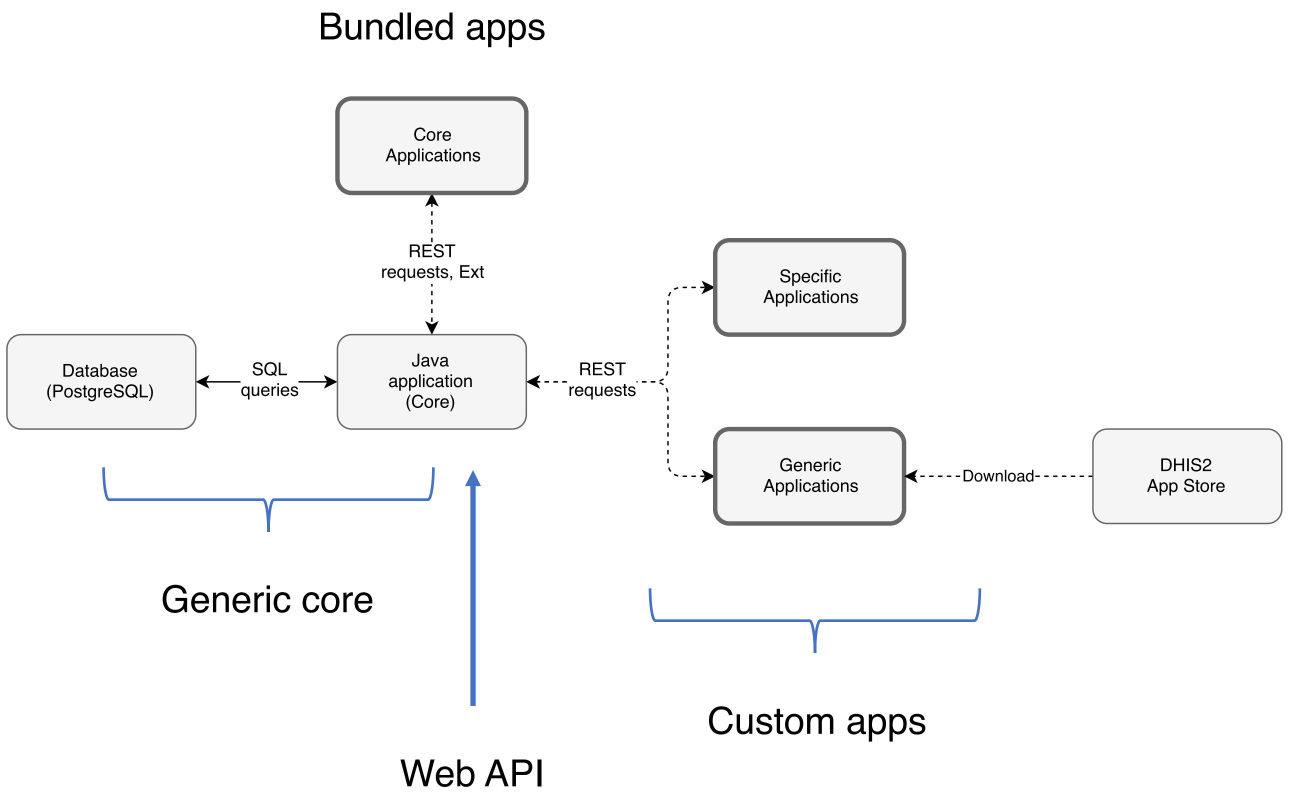
Outline

- About DHIS2 and its data structure
- Communicating with the web API
- Building an app and making it talk with DHIS2
- Installing an app in DHIS2

DHIS2 as a platform

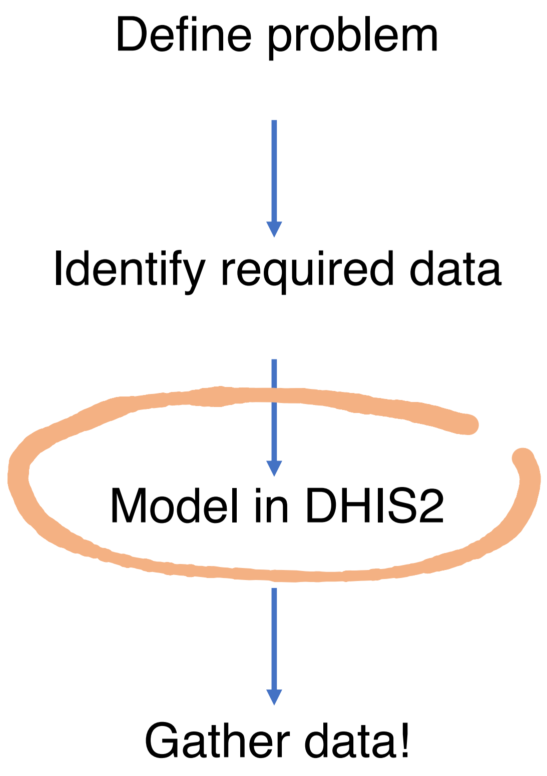


Roland, L. K., Sanner, T. A., Sæbø, J. I., & Monteiro, E. (2017). P for platform. architectures of large-scale participatory design. *Scandinavian Journal of Information Systems*, *29*(1).



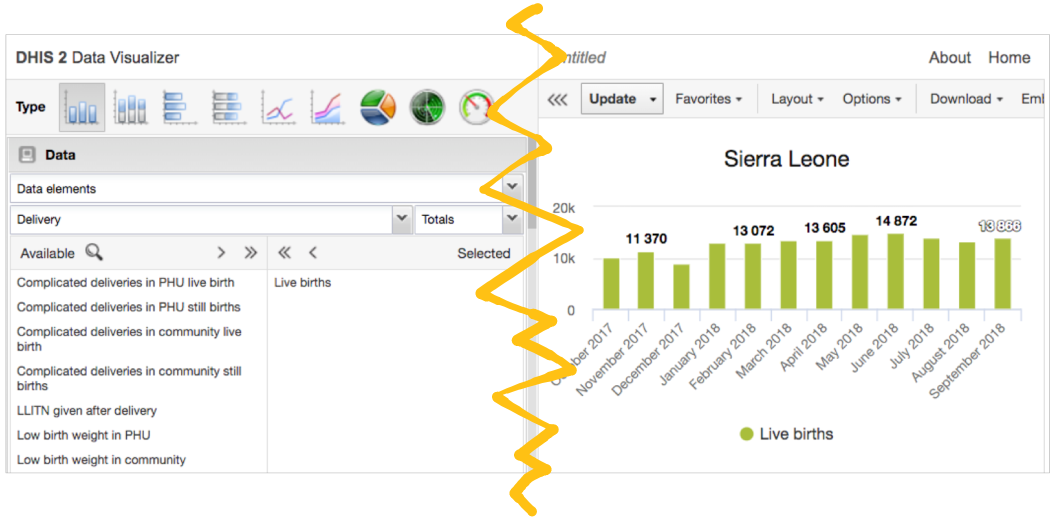
The flexible data structure

- DHIS has a flexible data structure
  - Adaptable to different contexts
  - Should be able to change the metadata model in a GUI
  - Key principle since the beginning



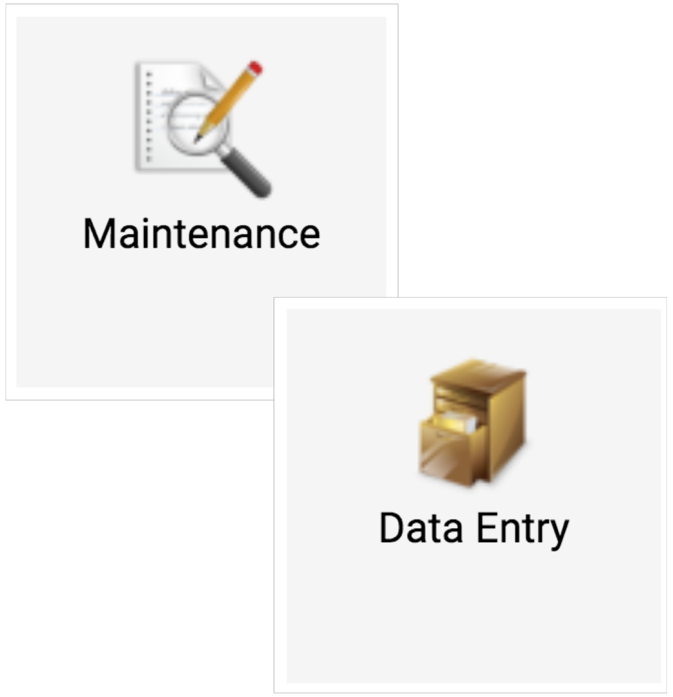
Metadata vs. collected data

- Important distinction between
  - Metadata
    - Description of data
    - Abstraction of the real world
    - Configured by *implementers*
  - Collected data
    - Entered by data clerks, doctors etc.
    - Using data entry apps (Tracker/Event Capture, Data Entry)



Available:
- Complicated deliveries in PHU live birth
- Complicated deliveries in PHU still births
- Complicated deliveries in community live birth
- Complicated deliveries in community still births
- LLITN given after delivery
- Low birth weight in PHU
- Low birth weight in community

Selected:
- Live births



Three data dimensions

The “what”, the “where” and the “when” of collected data values

What: Data element

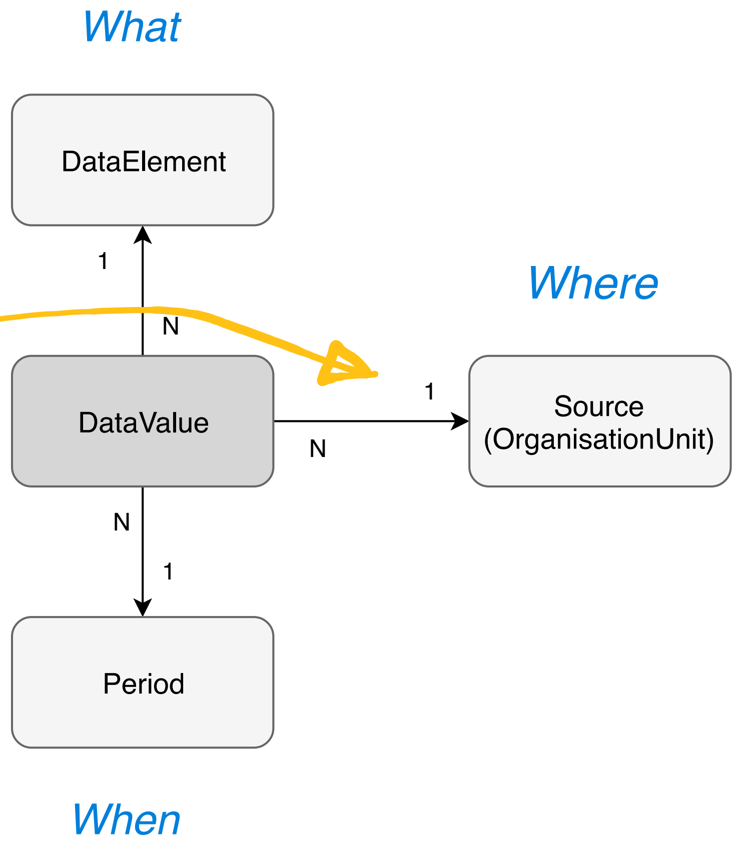
- What are we measuring?
- Mostly primitive types, e.g. number, string, boolean, date
- Example: Number of new measles cases, Age in years

Where: Organisation unit

- Where a health event took place, or a data value was collected
- Typically a health clinic, hospital etc.

When: Time period

- When the data value was collected (period or timestamp)
- Typically grouped by data presentation apps
  - (weekly, monthly, annually etc.)

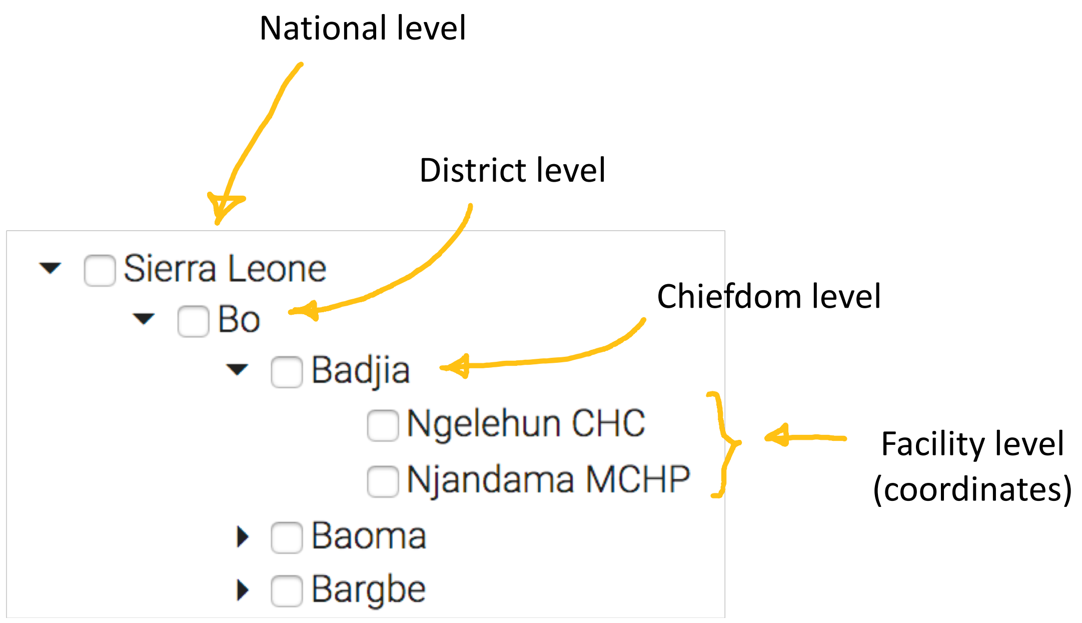


Organisation unit hierarchy

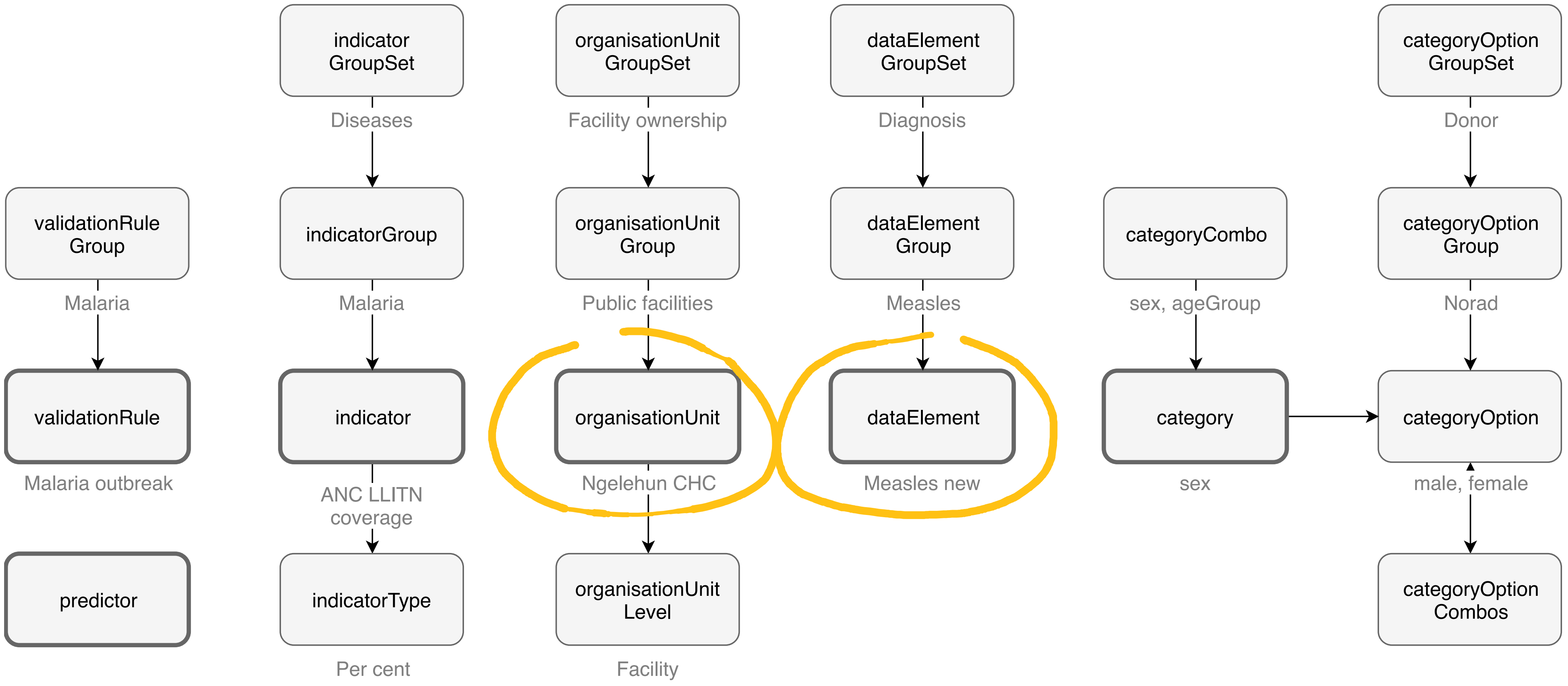
- Organisation units are structured in a tree-like hierarchy
- Organisation Units
  - Either a specific, geographical position (i.e. coordinates)
    - Health clinics, hospitals etc.
  - Or a geographical area (list of coordinates/polygon)
    - E.g. a country, region, city etc.

- Each unit has an organisation unit **level**
  - E.g. ”national”, “district”, “chiefdom” or “facility”
  - These are also user-defined

Sierra Leone

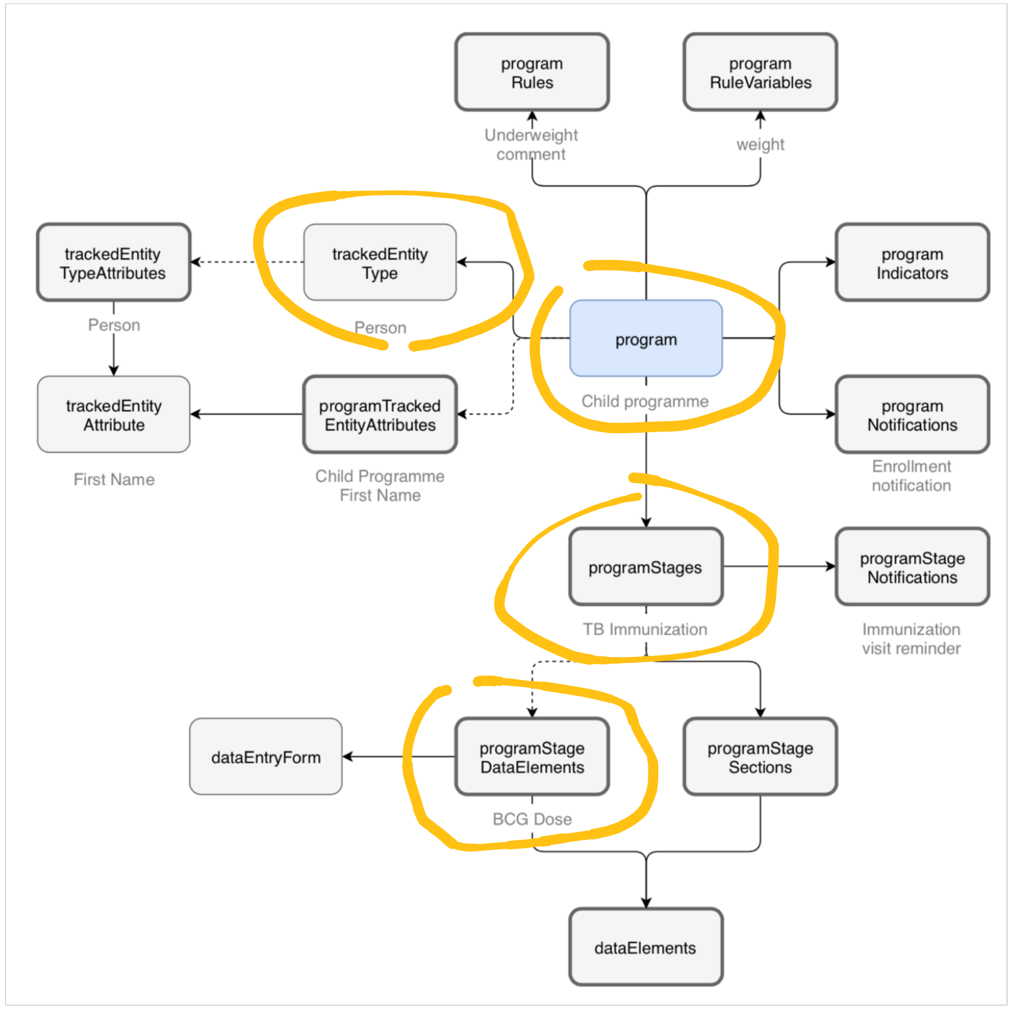


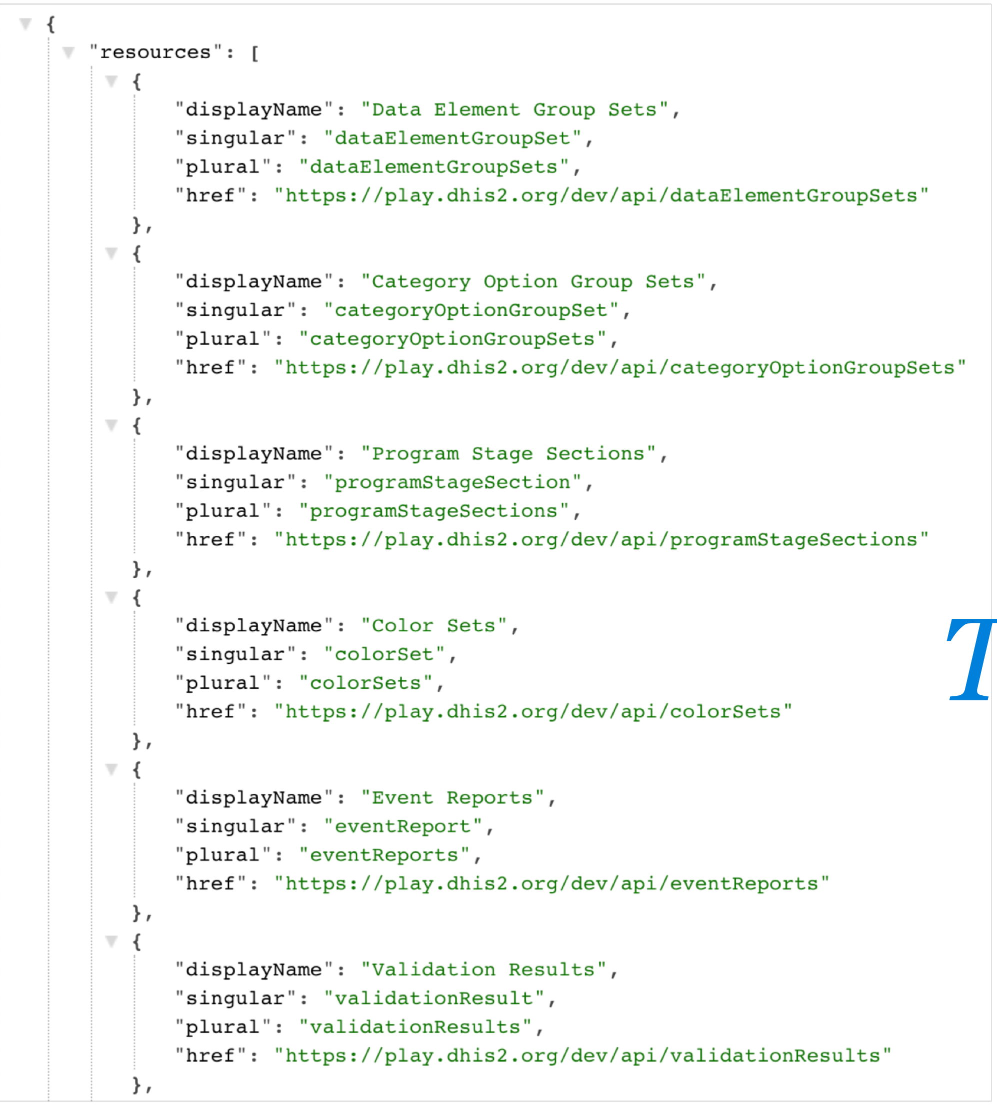
Some common metadata



“Aggregate” and Tracker

- Two conceptual ways of collecting data
- Aggregation
    - Great for collecting **routine** data sets
    - *Example*: Monthly reporting of a data set from health clinics
- Tracker
    - Great for capturing *processes* for a certain entity
    - The tracked entity can be a person, a health commodity, lab sample etc.
    - Either a chain of different stages or one, repeatable stage
    - Each stage has a number of data elements to collect
    - *Example*: Tracking a child through their vaccination program
        - Stage: Tubercolosis vaccination
            - Data element: BCG Dose





```
{
  "resources": [
    {
      "displayName": "Data Element Group Sets",
      "singular": "dataElementGroupSet",
      "plural": "dataElementGroupSets",
      "href": "https://play.dhis2.org/dev/api/dataElementGroupSets"
    },
    {
      "displayName": "Category Option Group Sets",
      "singular": "categoryOptionGroupSet",
      "plural": "categoryOptionGroupSets",
      "href": "https://play.dhis2.org/dev/api/categoryOptionGroupSets"
    },
    {
      "displayName": "Program Stage Sections",
      "singular": "programStageSection",
      "plural": "programStageSections",
      "href": "https://play.dhis2.org/dev/api/programStageSections"
    },
    {
      "displayName": "Color Sets",
      "singular": "colorSet",
      "plural": "colorSets",
      "href": "https://play.dhis2.org/dev/api/colorSets"
    },
    {
      "displayName": "Event Reports",
      "singular": "eventReport",
      "plural": "eventReports",
      "href": "https://play.dhis2.org/dev/api/eventReports"
    },
    {
      "displayName": "Validation Results",
      "singular": "validationResult",
      "plural": "validationResults",
      "href": "https://play.dhis2.org/dev/api/validationResults"
    },
```

The Web API

DHIS2's Java core

- An enormous Java monolith application (280k java lines across 2946 files)
    - All available on github: https://github.com/dhis2/dhis2-core
- Supplies the interface between the database and applications
    - Contains logic for the system components
- **Exposes the web API**
    - Based on the REST architecture

Accessing the API

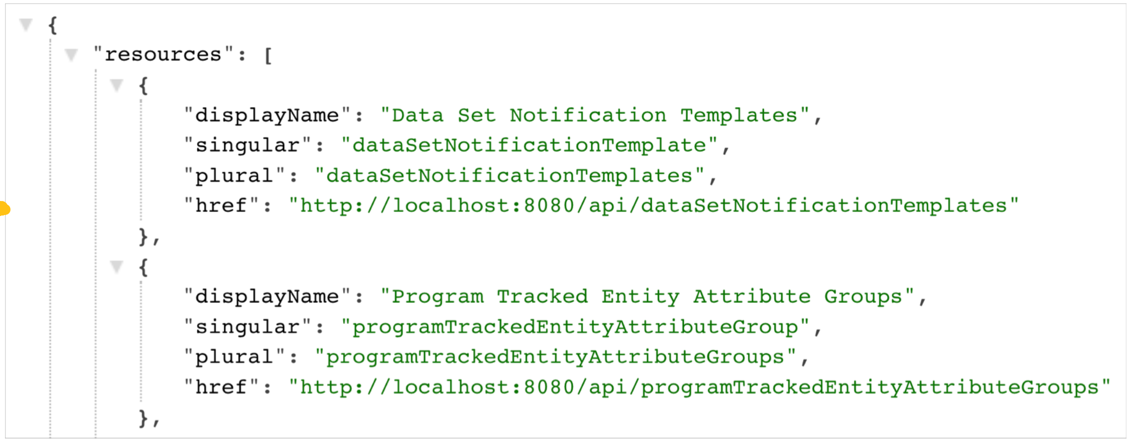
- In a browser, using a log-in session
  1. Navigate to the URL (`https://<dhis2-url>/api`)
  2. Log in with your credentials

- With an “Authorization” header
  - *`Basic <Base 64-encoded string>`*
  - Encoded string: `btoa(username:password)`
  - Example for `admin:district`; `Basic YWRtaW46ZGlzdHJpY3Q=`

Navigating the API

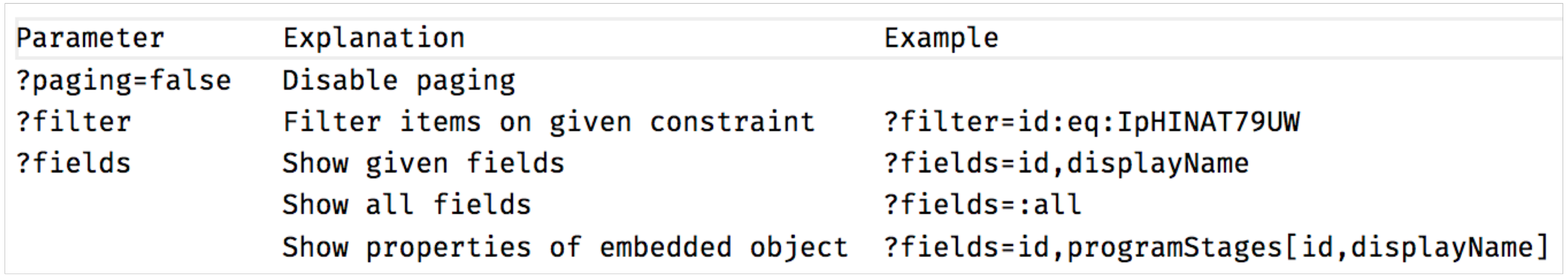
- Can be viewed in any web browser
  - With ordinary GET-requests
  - Returns *xml* by default, *json* with a .json suffix

- `/api/resources`
  - Contains a list of all **metadata** endpoints



```
{
  "resources": [
    {
      "displayName": "Data Set Notification Templates",
      "singular": "dataSetNotificationTemplate",
      "plural": "dataSetNotificationTemplates",
      "href": "http://localhost:8080/api/dataSetNotificationTemplates"
    },
    {
      "displayName": "Program Tracked Entity Attribute Groups",
      "singular": "programTrackedEntityAttributeGroup",
      "plural": "programTrackedEntityAttributeGroups",
      "href": "http://localhost:8080/api/programTrackedEntityAttributeGroups"
    },
```

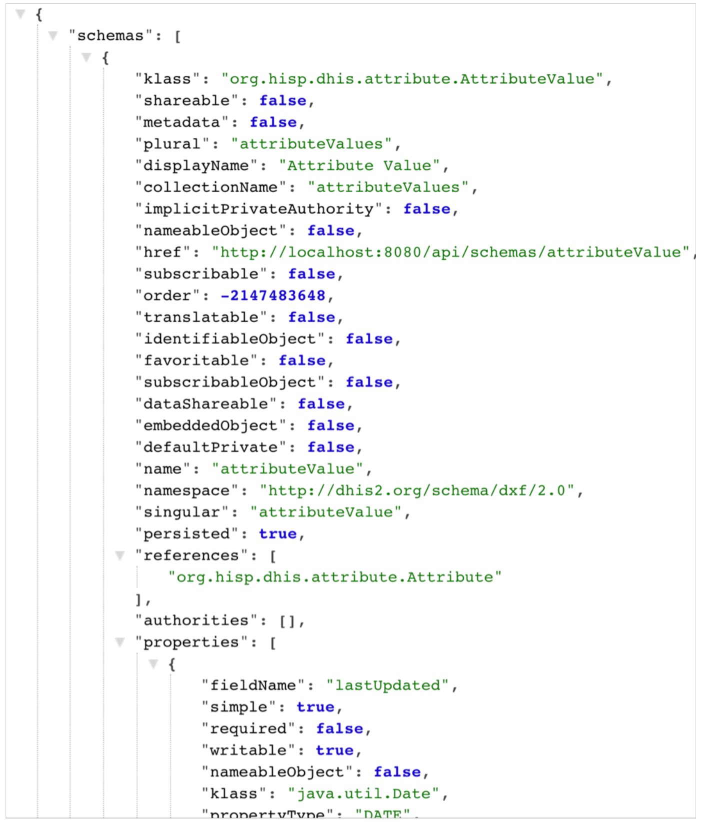
- `/api/<resource[s]>`
  - List all metadata items of a certain type
  - Available parameters:



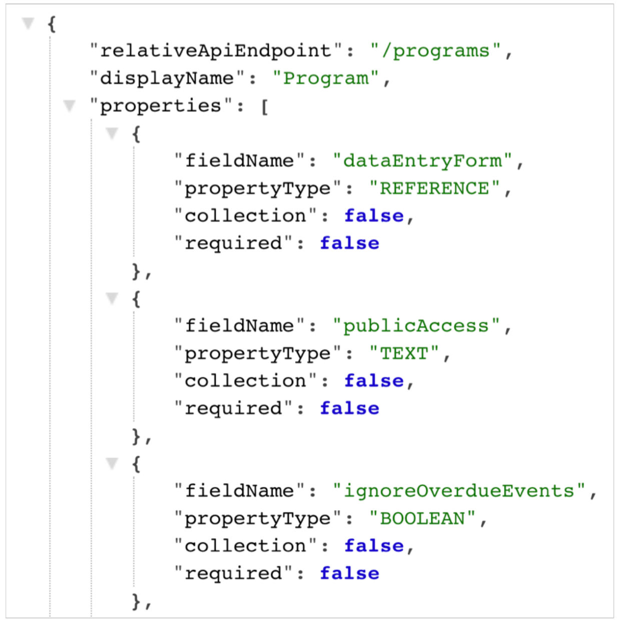
Parameter	Explanation	Example
?paging=false	Disable paging	
?filter	Filter items on given constraint	?filter=id:eq:IpHINAT79UW
?fields	Show given fields	?fields=id,displayName
	Show all fields	?fields=:all
	Show properties of embedded object	?fields=id,programStages[id,displayName]

Exploring resources with Schemas

- `/api/schemas`
  - Show key attributes for all available resources
- `/api/schemas/<resource>`
  - Show *all* attributes for one specific resource



```
{
  "schemas": [
    {
      "klass": "org.hisp.dhis.attribute.AttributeValue",
      "shareable": false,
      "metadata": false,
      "plural": "attributeValues",
      "displayName": "Attribute Value",
      "collectionName": "attributeValues",
      "implicitPrivateAuthority": false,
      "nameableObject": false,
      "href": "http://localhost:8080/api/schemas/attributeValue",
      "subscribable": false,
      "order": -2147483648,
      "translatable": false,
      "identifiableObject": false,
      "favoritable": false,
      "subscribableObject": false,
      "dataShareable": false,
      "embeddedObject": false,
      "defaultPrivate": false,
      "name": "attributeValue",
      "namespace": "http://dhis2.org/schema/dxf/2.0",
      "singular": "attributeValue",
      "persisted": true,
      "references": [
        "org.hisp.dhis.attribute.Attribute"
      ],
      "authorities": [],
      "properties": [
        {
          "fieldName": "lastUpdated",
          "simple": true,
          "required": false,
          "writable": true,
          "nameableObject": false,
          "klass": "java.util.Date",
          "propertyType": "DATE",
```



```
{
  "relativeApiEndpoint": "/programs",
  "displayName": "Program",
  "properties": [
    {
      "fieldName": "dataEntryForm",
      "propertyType": "REFERENCE",
      "collection": false,
      "required": false
    },
    {
      "fieldName": "publicAccess",
      "propertyType": "TEXT",
      "collection": false,
      "required": false
    },
    {
      "fieldName": "ignoreOverdueEvents",
      "propertyType": "BOOLEAN",
      "collection": false,
      "required": false
    },
```

Modifying data

- API supports the following other methods:
  - **POST**
    - Creates a new entry
    - See resource schema on `/api/schemas/<resource>` for required fields
  - **DELETE**
    - Delete an entry
    - Might have dependencies!
  - **PUT**
    - *Replace* the whole item
    - Requires app to download whole object
  - **PATCH**
    - Change specific attributes
    - Might not work on all endpoints – try!

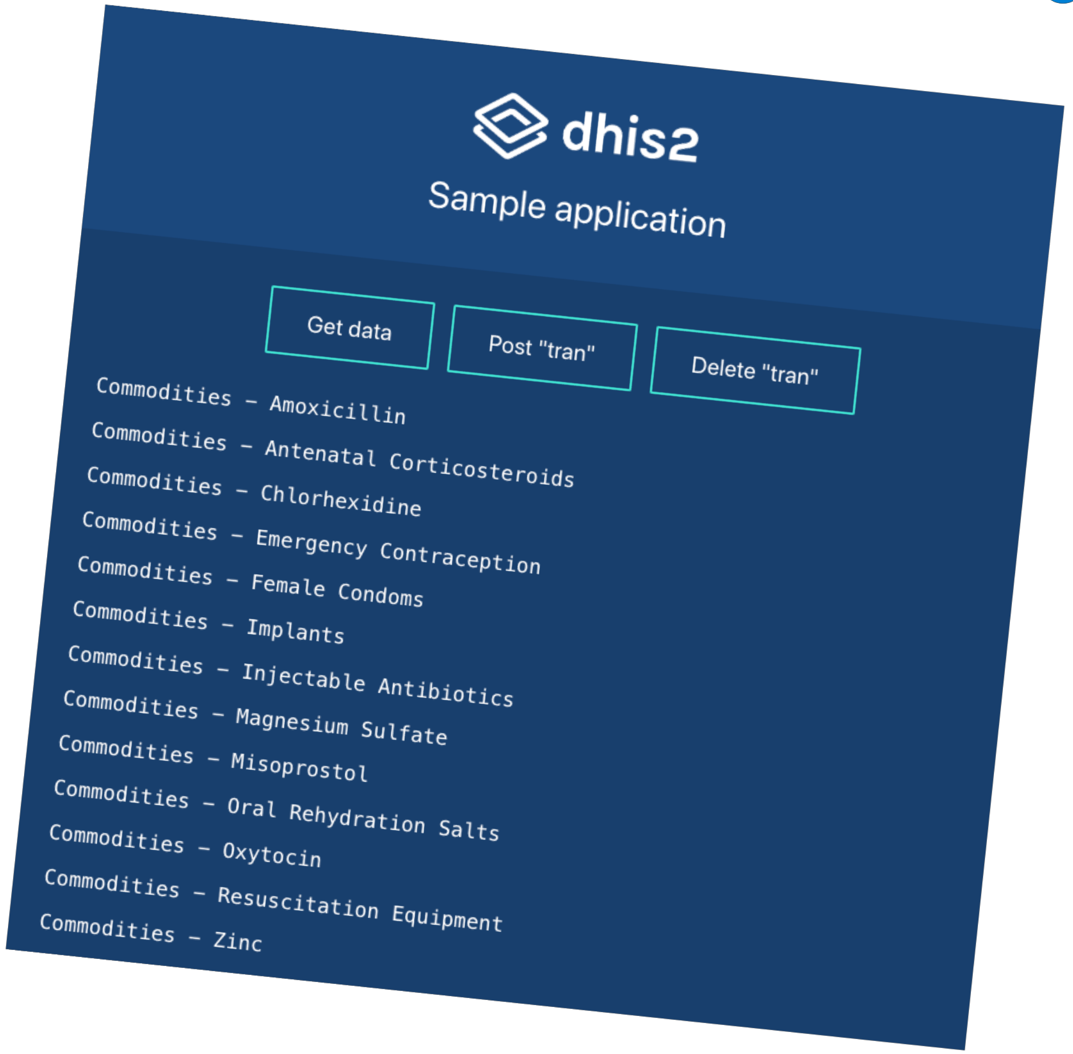
Prototype with Postman, Curl or a similar tool!

POSTMAN

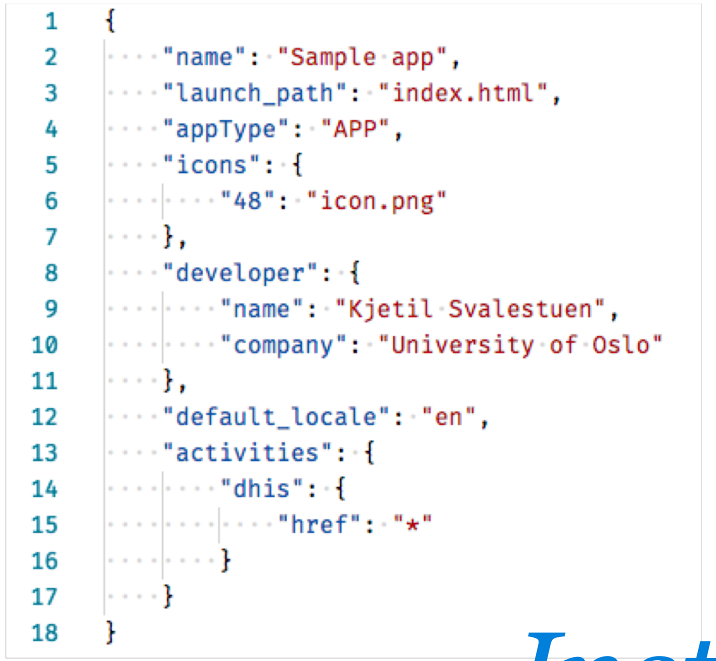
Let’s build an <App />!

Agenda for this session

1. Create a basic React application
2. Fill it with some data
3. Post new data
4. Delete existing data
5. Installing your app in DHIS2



Commodities – Amoxicillin
Commodities – Antenatal Corticosteroids
Commodities – Chlorhexidine
Commodities – Emergency Contraception
Commodities – Female Condoms
Commodities – Implants
Commodities – Injectable Antibiotics
Commodities – Magnesium Sulfate
Commodities – Misoprostol
Commodities – Oral Rehydration Salts
Commodities – Oxytocin
Commodities – Resuscitation Equipment
Commodities – Zinc



```
{
    "name": "Sample app",
    "launch_path": "index.html",
    "appType": "APP",
    "icons": {
        "48": "icon.png"
    },
    "developer": {
        "name": "Kjetil Svalestuen",
        "company": "University of Oslo"
    },
    "default_locale": "en",
    "activities": {
        "dhis": {
            "href": "*"
        }
    }
}
```

Installing the app in DHIS2

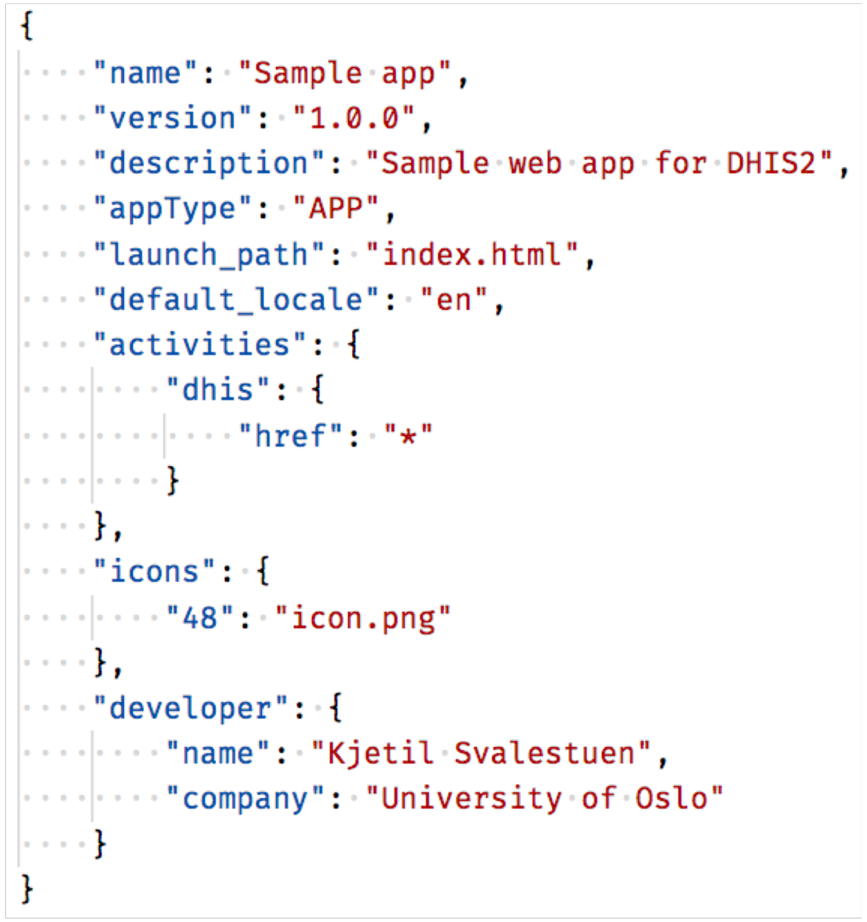
The manifest file

`manifest.webapp`

- Place at root level in your bundled application
- Tells DHIS2 about important properties of your app
- Can be generated using the *d2-manifest* NPM package

`activities.dhis.href: *`

- Converted to URL of DHIS2 instance
- Read manifest file **in production** to get the URL
    - Use webpack with NODE_ENV



```json
{
    "name": "Sample app",
    "version": "1.0.0",
    "description": "Sample web app for DHIS2",
    "appType": "APP",
    "launch_path": "index.html",
    "default_locale": "en",
    "activities": {
        "dhis": {
            "href": "*"
        }
    },
    "icons": {
        "48": "icon.png"
    },
    "developer": {
        "name": "Kjetil Svalestuen",
        "company": "University of Oslo"
    }
}
```

Some words of advice

1. Let the backend do the hard work
   - Use the `filter` and `fields` parameters for what they're worth

2. If you're suddenly facing a wall of errors, you might be logged out
   - Refresh your login token by visiting the login-page

3. Use your browser and other tools like Postman or Curl
   - Might be easier than debugging the API through your app

4. Consider hosting your own DHIS2 instance
   - Chore to configure, but comes with a few benefits
     1. Nobody will mess up your data (except for yourself)
     2. Access to server logs and error stacks