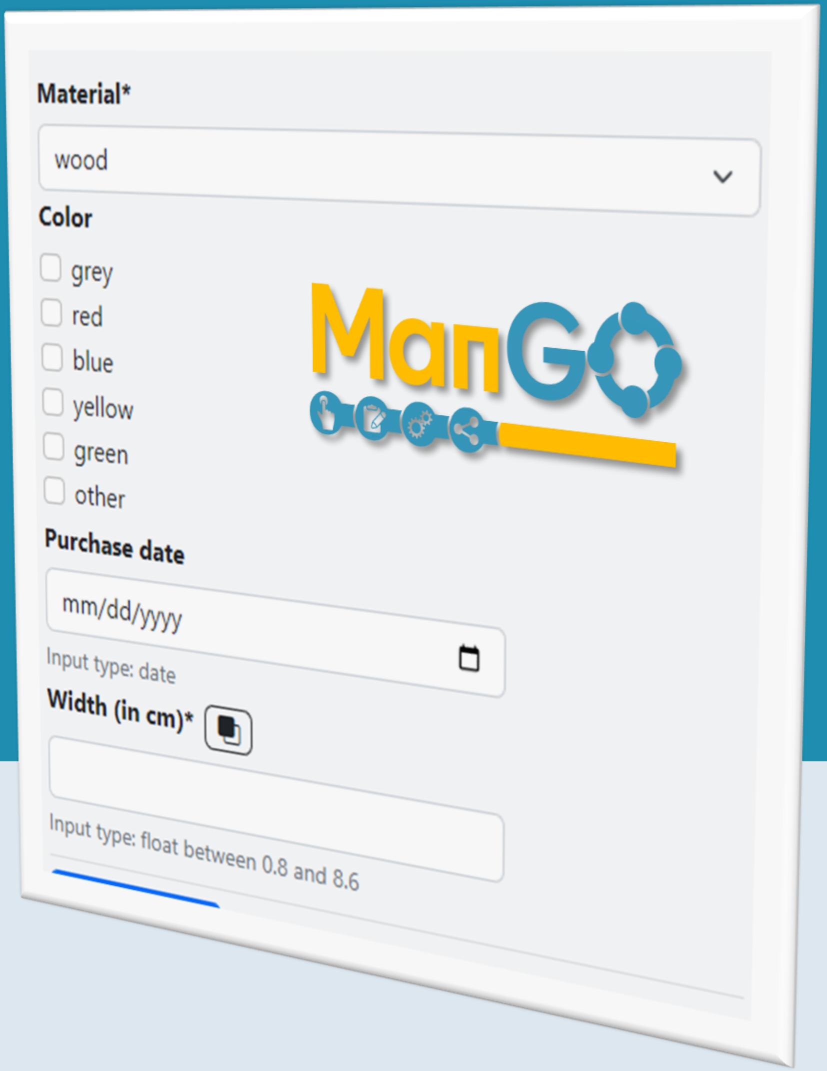
KU LEUVEN

Towards rich and standardized metadata in iRODS



Material*

wood

Color

□ grey
□ red
□ blue
□ yellow
□ green
□ other

Purchase date

mm/dd/yyyy

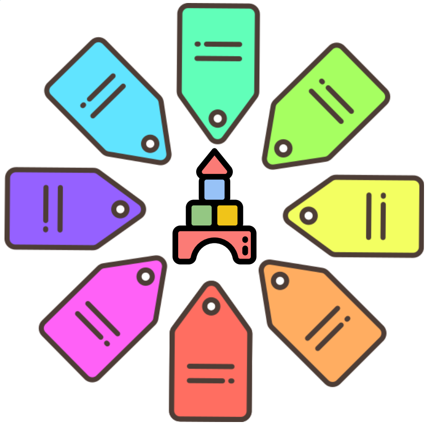
Input type: date

Width (in cm)*

Input type: float between 0.8 and 8.6

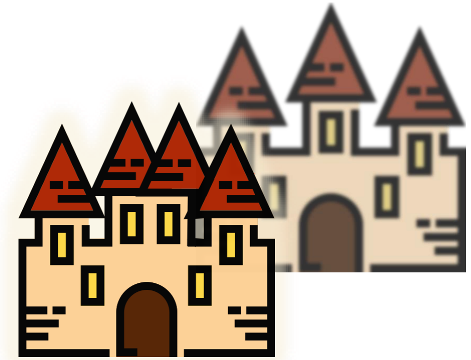
Mariana Montes & Paul Borgermans

Metadata schemas in the ManGO portal



Apply rich and standardized metadata

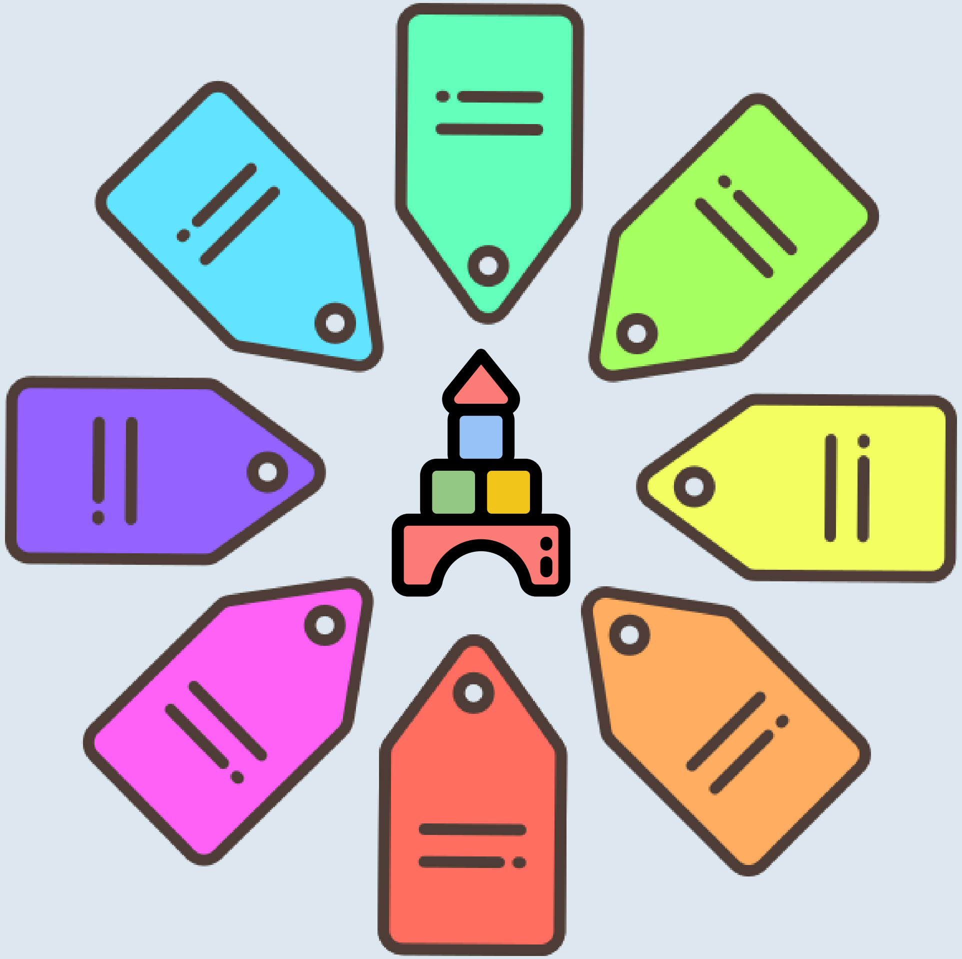
Hierarchical structure for metadata



Schema life cycle

First challenge

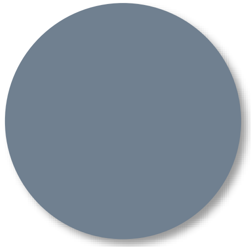
Applying rich, standardized metadata in a systematic way

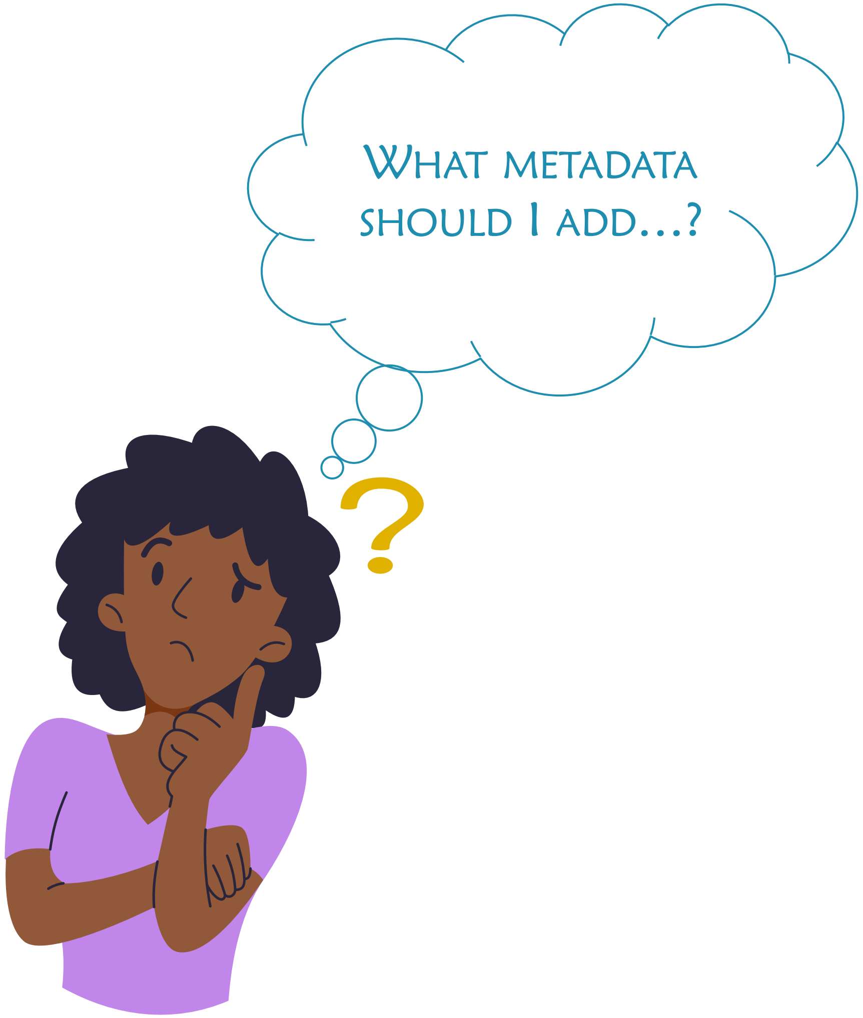


Metadata without schemas

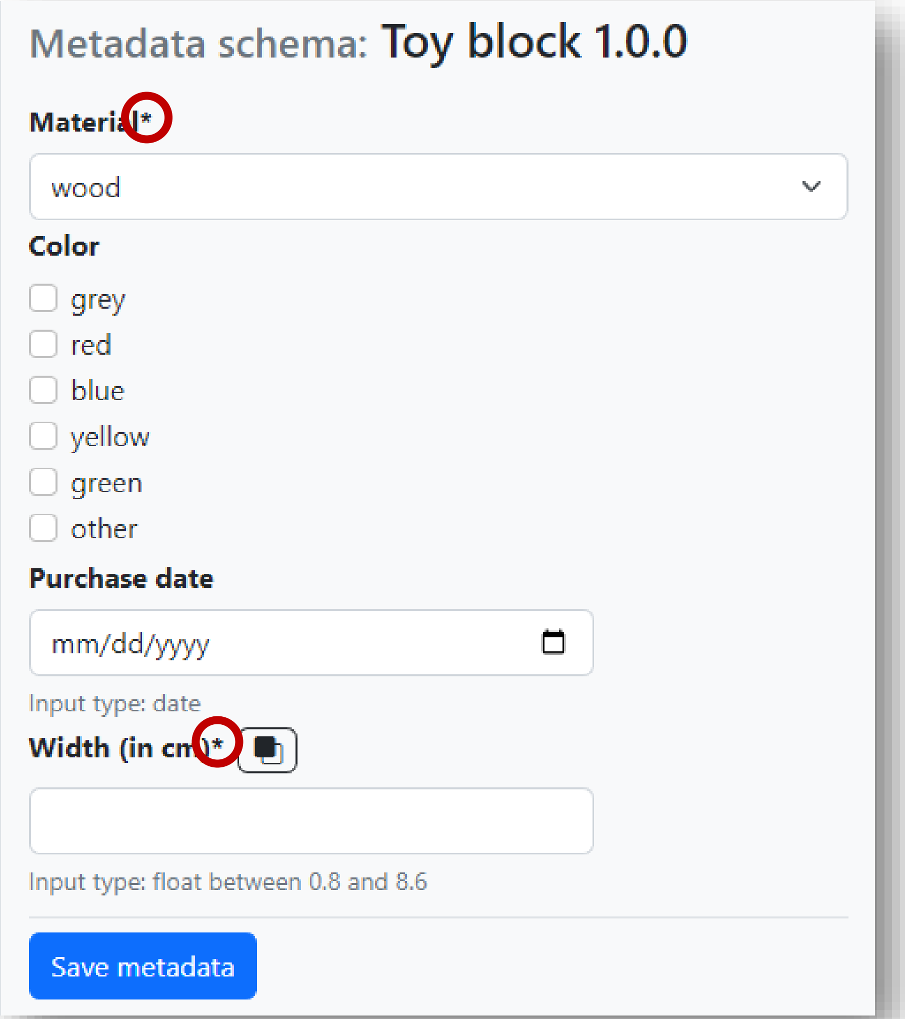
Typos, inconsistencies…

```
color: gray
colour: grey
COLOR: Gray
color: gris
kleur: grijs
colour: slate gray
colour: SlateGray
colour: slat gray
colour: #708090
```





The alternative: metadata with schemas (forms)



Metadata schema: **Toy block 1.0.0**

Material*

wood

Color

□ grey
□ red
□ blue
□ yellow
□ green
□ other

Purchase date

mm/dd/yyyy

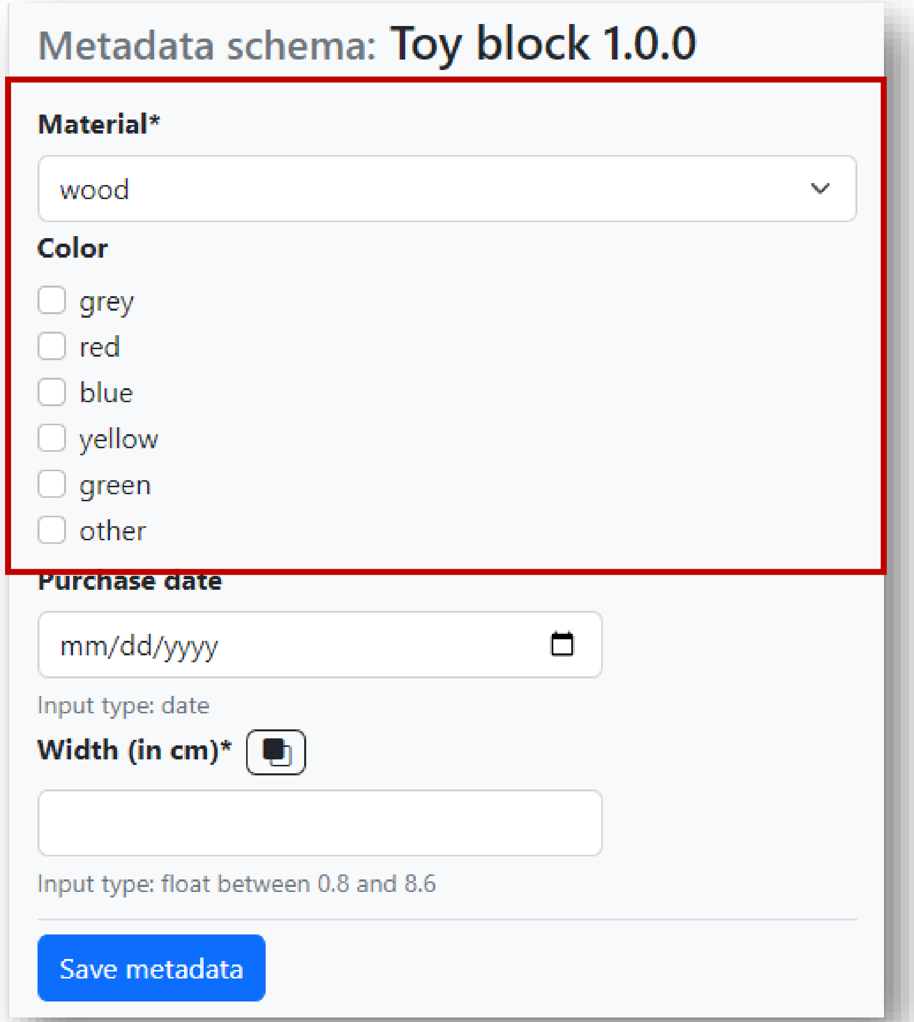
Input type: date

Width (in cm)*

Input type: float between 0.8 and 8.6

- All necessary and suggested fields.

The alternative: metadata with schemas (forms)



Metadata schema: Toy block 1.0.0

Material*

wood

Color

□ grey
□ red
□ blue
□ yellow
□ green
□ other

Purchase date

mm/dd/yyyy

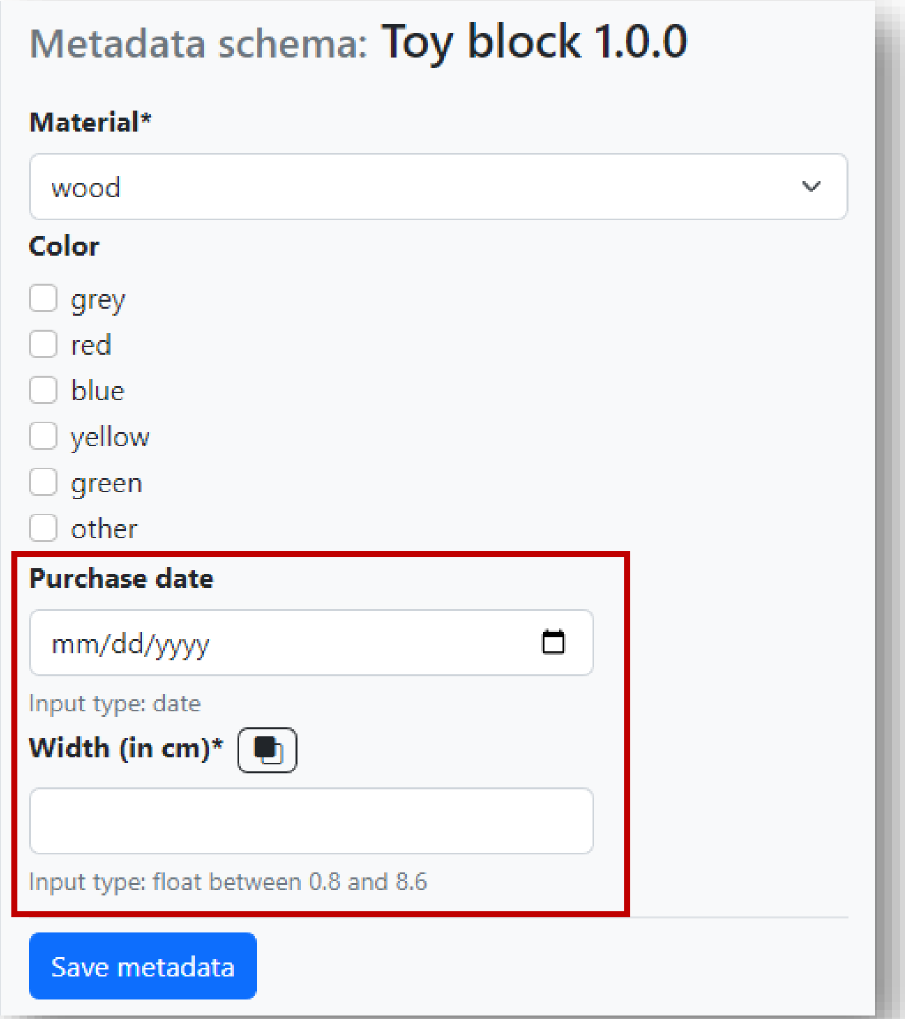
Input type: date

Width (in cm)*

Input type: float between 0.8 and 8.6

- All necessary and suggested fields.
- Predefined options for the values.

The alternative: metadata with schemas (forms)



Metadata schema: Toy block 1.0.0

Material*

wood

Color

□ grey
□ red
□ blue
□ yellow
□ green
□ other

Purchase date

mm/dd/yyyy

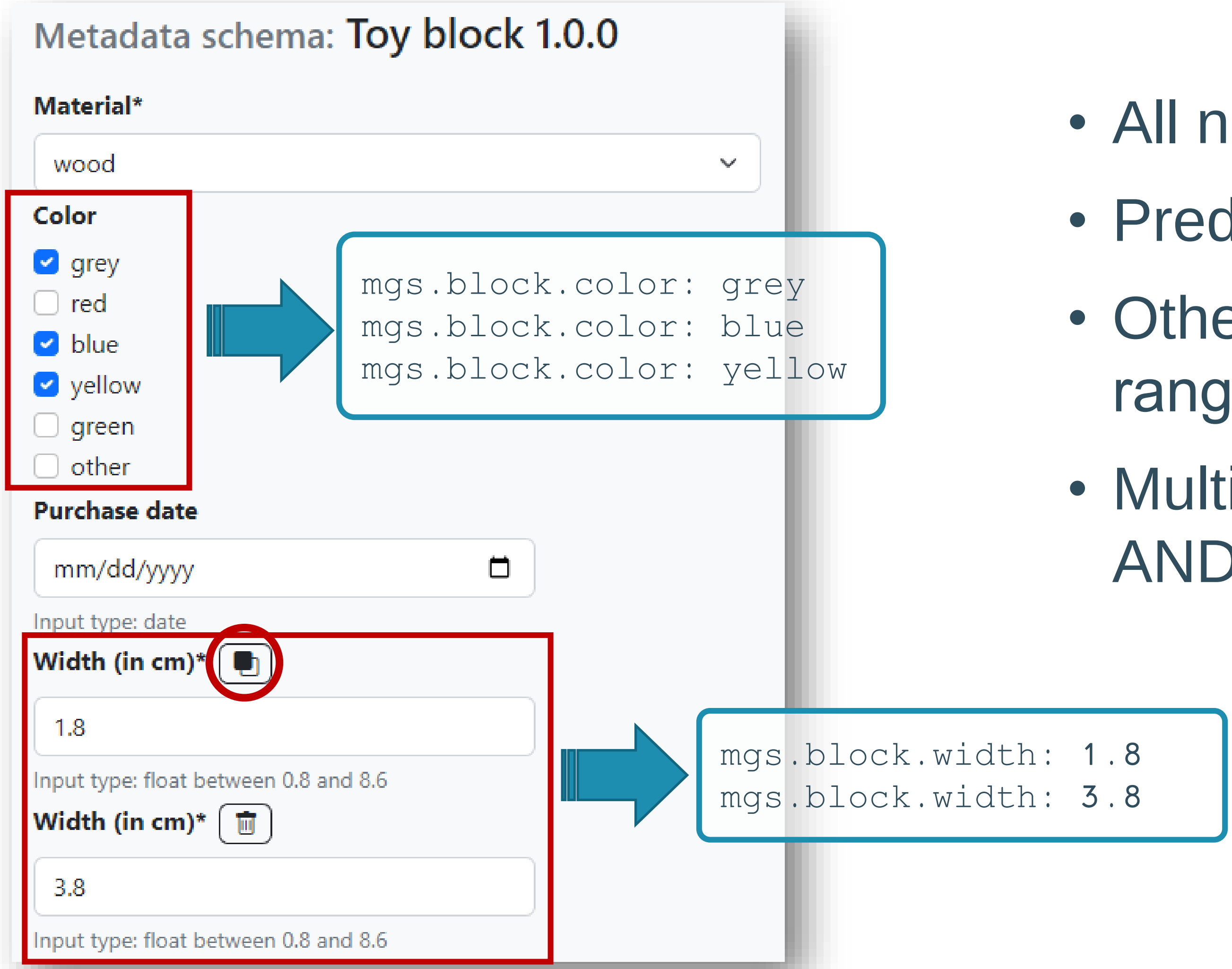
Input type: date

Width (in cm)*

Input type: float between 0.8 and 8.6

- All necessary and suggested fields.
- Predefined options for the values.
- Other types of validation (formats, ranges…)

The alternative: metadata with schemas (forms)



Metadata schema: Toy block 1.0.0

Material*

wood

Color

- ☑ grey
- ☐ red
- ☑ blue
- ☑ yellow
- ☐ green
- ☐ other

```
mgs.block.color: grey
mgs.block.color: blue
mgs.block.color: yellow
```

Purchase date

mm/dd/yyyy

Input type: date

Width (in cm)*

1.8

Input type: float between 0.8 and 8.6

Width (in cm)*

3.8

Input type: float between 0.8 and 8.6

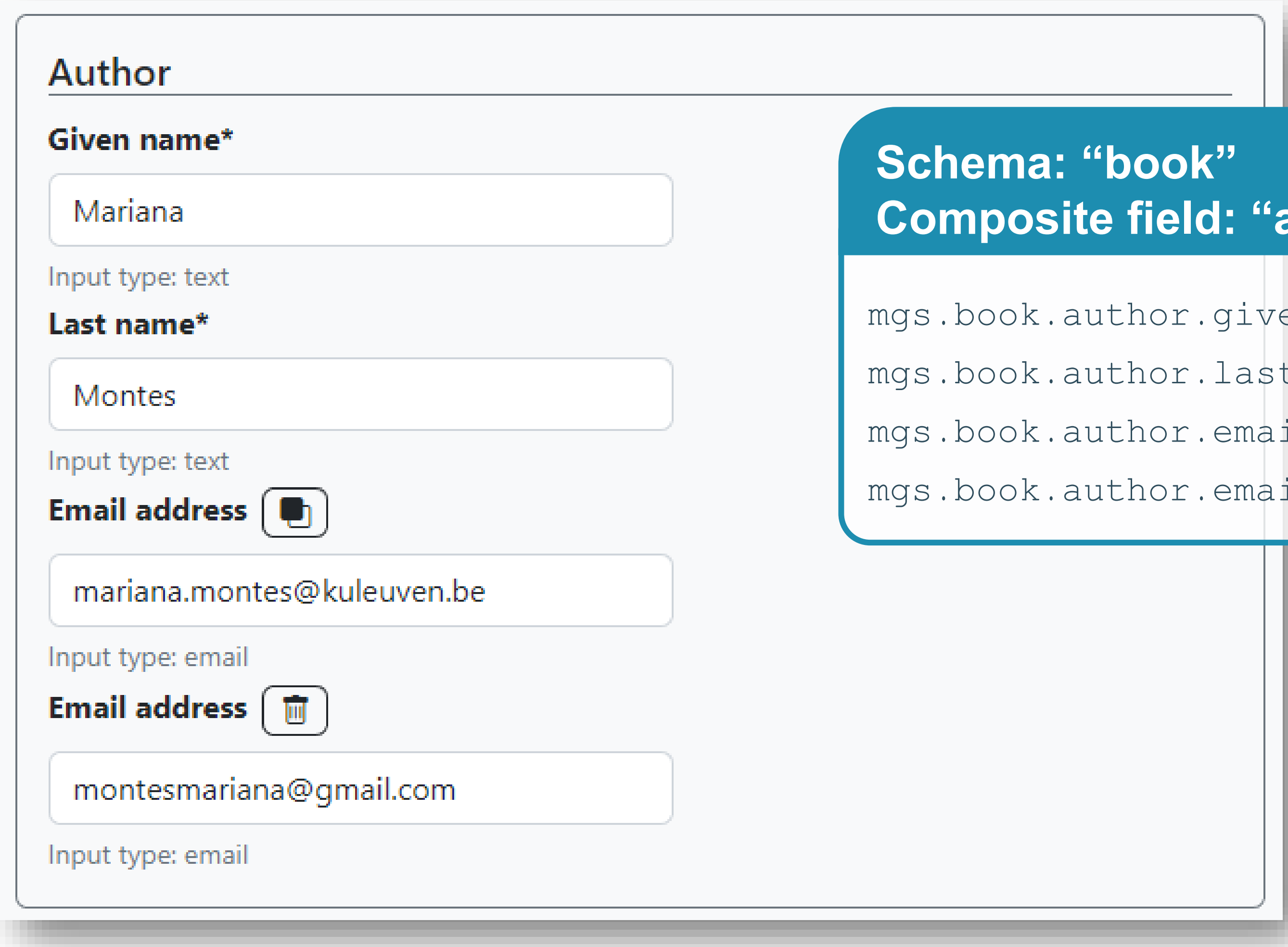
```
mgs.block.width: 1.8
mgs.block.width: 3.8
```

- All necessary and suggested fields.
- Predefined options for the values.
- Other types of validation (formats, ranges…)
- Multiple AVUs with the same name AND consistency in names.

Second challenge

Metadata with a hierarchical structure

Hierarchical structure with schemas: namespacing



Author

Given name*

Mariana

Input type: text

Last name*

Montes

Input type: text

Email address

mariana.montes@kuleuven.be

Input type: email

Email address

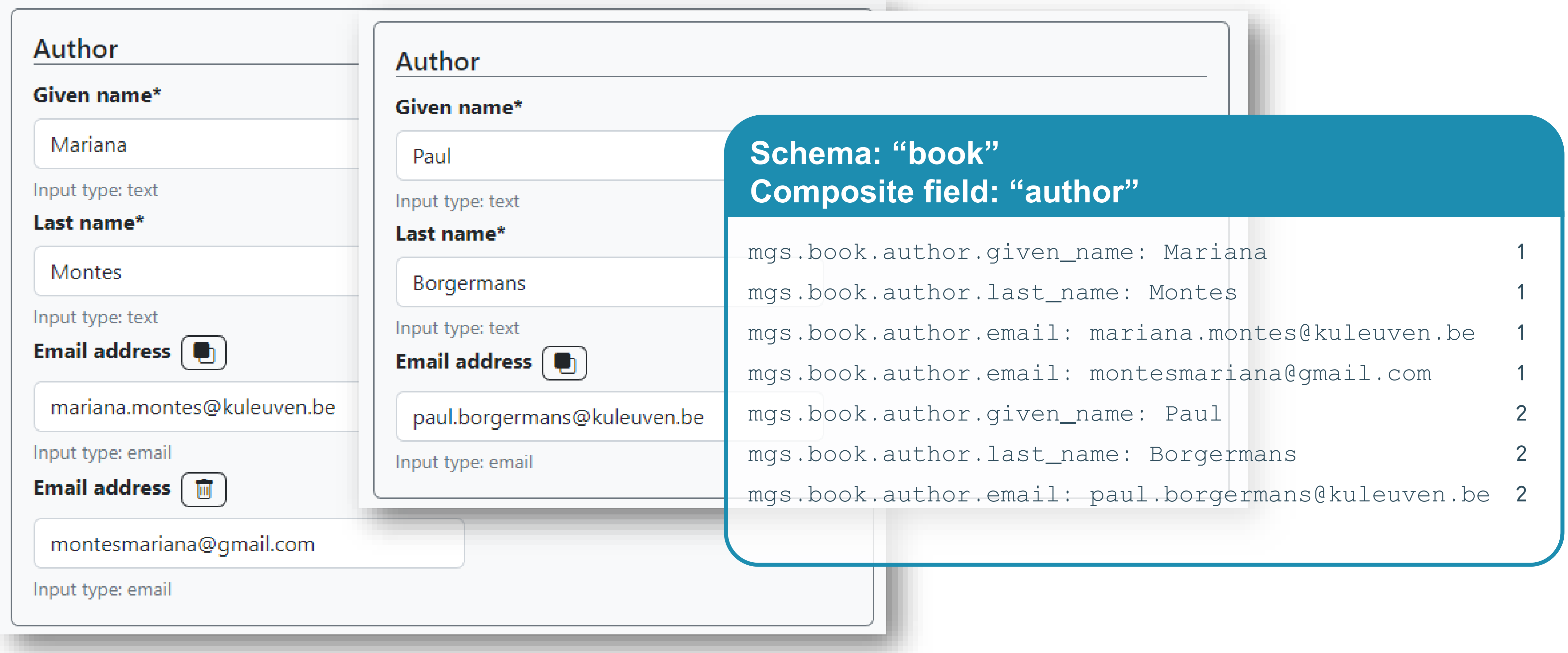
montesmariana@gmail.com

Input type: email

Schema: “book”
Composite field: “author”

```
mgs.book.author.given_name: Mariana
mgs.book.author.last_name: Montes
mgs.book.author.email: mariana.montes@kuleuven.be
mgs.book.author.email: montesmariana@gmail.com
```

Hierarchical structure with schemas: units



Author

Given name*

Mariana

Input type: text

Last name*

Montes

Input type: text

Email address ⧉

mariana.montes@kuleuven.be

Input type: email

Email address 🗑

montesmariana@gmail.com

Input type: email

Author

Given name*

Paul

Input type: text

Last name*

Borgermans

Input type: text

Email address ⧉

paul.borgermans@kuleuven.be

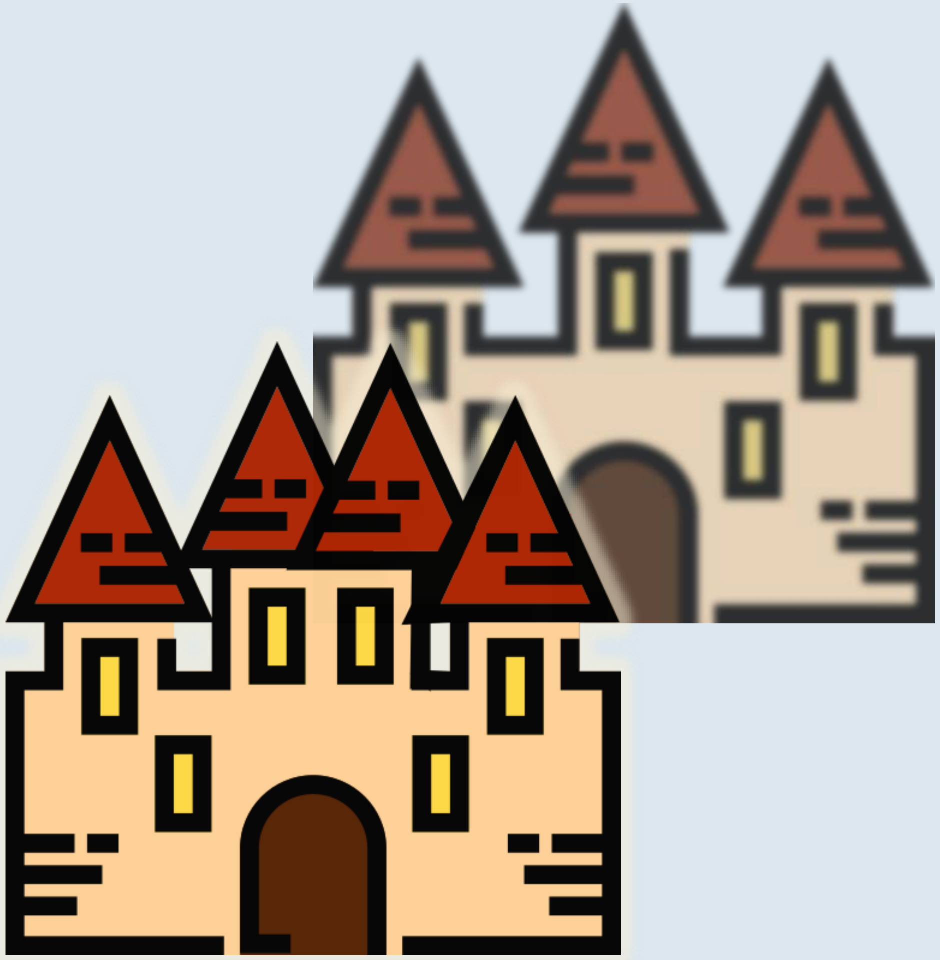
Input type: email

Schema: “book”
Composite field: “author”

```
mgs.book.author.given_name: Mariana                    1
mgs.book.author.last_name: Montes                      1
mgs.book.author.email: mariana.montes@kuleuven.be      1
mgs.book.author.email: montesmariana@gmail.com         1
mgs.book.author.given_name: Paul                       2
mgs.book.author.last_name: Borgermans                  2
mgs.book.author.email: paul.borgermans@kuleuven.be     2
```

Third challenge

Combining stability and evolution potential of metadata schemas



Without life cycle

Every schema editable AND implementable

Each schema editable OR implementable, new schemas every time

Without life cycle

UNRELIABLE

y schema editable
AND implementable

Each schema editable
OR implementable,
new schemas every
time

Without life cycle

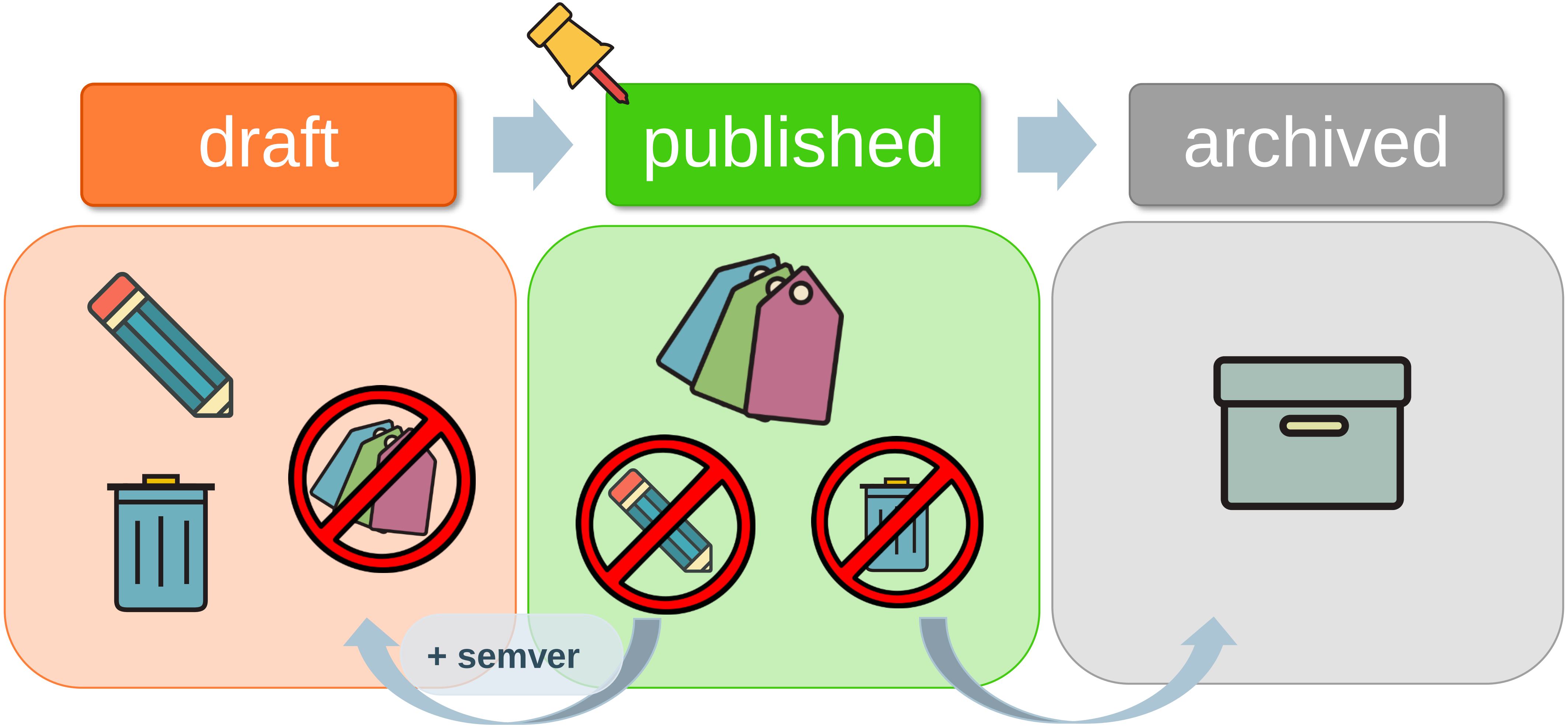
UNRELIABLE

y schema editable
AND implementable

NO HISTORY

editable
mplementable,
new schemas every
time

With life cycle



draft

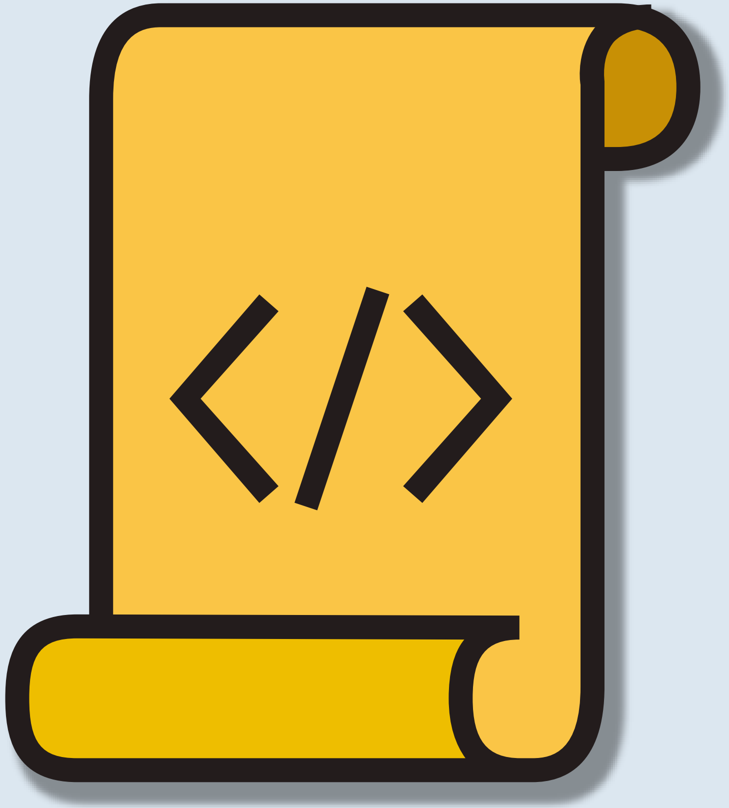
published

archived

+ semver

Technical specifications

Metadata schema manager under the hood



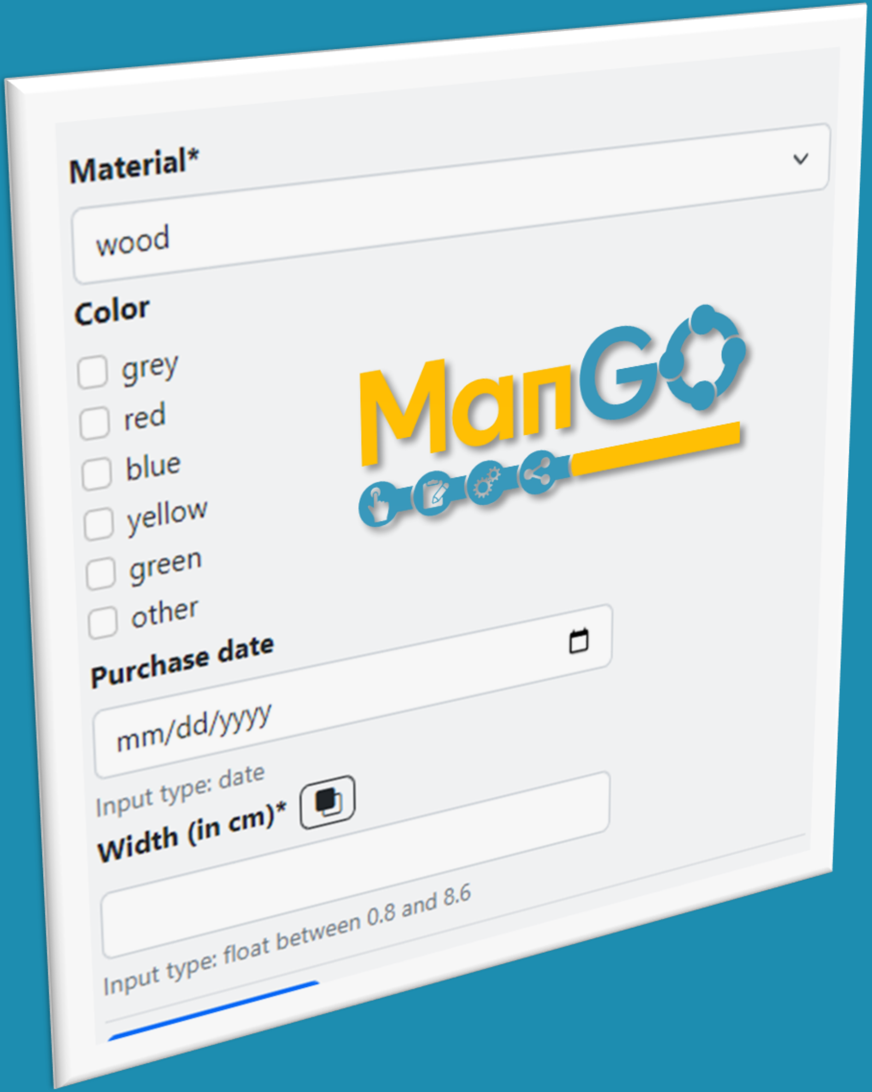
ManGO metadata schemas under the hood

Python in the backend,
JS in the frontend

Bootstrap 5.2
Forms and validation

```
{
  “schema_name”: “book”,
  “version”: “1.0.0”,
  “status”: “published”,
  “title”: “Book information”,
  “properties”: {
    “name”: {
      “type”: “text”,
      “title”: “Book title”
    }
  }
}
```

(Custom) JSON format



Material*

wood

Color

□ grey
□ red
□ blue
□ yellow
□ green
□ other

Purchase date

mm/dd/yyyy

Input type: date

Width (in cm)*

Input type: float between 0.8 and 8.6

Thank you!

mariana.montes@kuleuven.be
paul.borgermans@kuleuven.be

DEMO:
https://github.com/kuleuven/mango-metadata-schemas

JSON format: schema

```
{
  “schema_name”: “book”,
  “version”: “1.0.0”,
  “status”: “published”,
  “title”: “Book information”,
  “properties”: {…},
  “edited_by”: “username”,
  “realm”: “project_collection”,
  “parent”: “”
}
```

- Unique name
- Life cycle (versioning)
- Display name
- Components
- Other metadata

JSON format: simple field

```
“title”: {
  “type”: “text”,
  “title”: “Book title”,
  “required”: true,
  “help”: “Title in the book
(also translations)”,
  “repeatable”: true
}
```

Common fields

- Unique name
- Display name
- Type of field
- Boolean values: required, repeatable
- Help text, default value

JSON format: multiple-choice

```
“ebook”: {
  “type”: “select”,
  “title”: “Is there an e-book?”,
  “multiple”: false,
  “ui”: “radio”,
  “required”: true,
  “values”: [
    “Available”,
    “Unavailable”
  ]
}
```

Specific fields

- Whether only one or multiple values can be chosen
- How to render (dropdown, checkbox, radio)
- Array of possible values

JSON format: composite field

```
“author”: {
  “type”: “object”,
  “title”: “Author”,
  “properties”: {
    “name”: {
      “type”: “text”,
      “title”: “Name and surname”,
      “required”: true
    }
  },
  “repeatable”: true
}
```