JHipster Example for Human-Readable Foreign Keys in User Interface

This code was generated using the JHipster blueprint "generator-jhipster-multiple-human-readable-foreign-key-fields" (the source code is available at: https://github.com/amarpatel-xx/generator-jhipster-multiple-human-readable-foreign-key-fields). This code has a JDL which shows 2 foreign keys that will concatenated and shown, in the Angular user interface, in replacement of the UUID. The JDL can be modified and the @customAnnotation("DISPLAY_IN_GUI_RELATIONSHIP_LINK") can be used with any fields of an entity which would make it easier to identify that entity when displayed (as part of a relationship). Sometimes having a UUID makes it difficult for the human in the loop to figure out what the entity on a relationship's other side actually is. If multiple entity fields are necessary to replace the UUID, the fields can be delimmited via a specified delimiter using a @customAnnotation, as well (see the example JDL file included as part of this project).

1. Below is the example using the @customAnnotation and specifying the delimiter also.

```
entity Blog {
  @customAnnotation("DISPLAY_IN_GUI_RELATIONSHIP_LINK") @customAnnotation("-") name
String required minlength(3)
  @customAnnotation("DISPLAY_IN_GUI_RELATIONSHIP_LINK") @customAnnotation(" ") handle
String required minlength(2)
}

entity Post {
  title String required
  content TextBlob required
  date Instant required
}

relationship ManyToOne {
  Blog{user(login)} to User
  Post{blog} to Blog
}
```

Prerequisites:

- Java 11+

- Node.js 16+
- Docker Desktop
- JHipster 7.9.3

Table of Contents

Build Java Microservices using the Multiple Human-readable Foreign Key Fields Blueprint

1. To generate a microservices architecture with human-readable foreign key fields support, run the following command:

```
sh generate-code-non-reactive-mf.sh
```

2. You should see the message:

```
Congratulations, JHipster execution is complete!
Sponsored with ❤️  by @oktadev.
```

Run your Multiple Human-readable Foreign Key Fields Example

1. When the process is complete, cd into the `gateway` directory and start Keycloak and Eureka using Docker Compose.

```
cd gateway
docker compose -f src/main/docker/keycloak.yml up -d
docker compose -f src/main/docker/jhipster-registry.yml up -d
```

2. Start `gateway` database with Docker by opening a terminal and navigating to its directory and running the Docker command. Then start the `gateway` by running the Maven command.

```
npm run docker:db:up
./mvnw spring-boot:run
```

3. Start `blog` database with Docker by opening a terminal and navigating to its directory and running the Docker command. Then, start the `blog` microservice.

```
cd blog
npm run docker:db:up
./mvnw spring-boot:run
```

4. Start `store` database with Docker by opening a terminal and navigating to its directory and running the Docker command. Then, start the `store` microservice.

```
cd store
npm run docker:db:up
./mvnw spring-boot:run
```

Switch Identity Providers

JHipster ships with Keycloak when you choose OAuth 2.0 / OIDC as the authentication type.

If you'd like to use Okta for your identity provider, see JHipster's documentation.

TIP

You can configure JHipster quickly with the Okta CLI:

```
okta apps create jhipster
```

See the Code in Action

Now you can open your favorite browser to http://localhost:8080, and log in with the credentials displayed on the page.

Then create a Blog

1. Open your favorite browser to http://localhost:8080, and log in with the credentials displayed on the page.
2. Then, add a blog by giving it a name, handle and selecting a user.
3. Add a tag by giving it a name.
4. Finally, create a post by giving it a title, content, selecting a blog and a tag.

Notice the Blog column shows <blog-name>-<blog-handle> and not the UUID of the blog. That is success!

Have Fun with Micro Frontends and JHipster!

I hope you enjoyed this demo, and it helped you understand how to build better microservice architectures with human-readable foreign key fields.

Find the code on GitHub: https://github.com/amarpatel-xx/jhipster-multiple-human-readable-foreign-key-fields-example

Read the following blog post, by Matt Raible, that was used as inspiration for this project: Micro Frontends for Java Microservices