JDL

Introduction

JHipster Domain Language, also known as JDL is a database-related technology and an integral part of the JHipster ecosystem, a development platform, used for defining entities, their relationships and attributes. With JDL, developers can lay down the foundation for the application's architecture, through simple yet expressive syntax which facilitates productivity amongst developers and collaboration within a team.

Features

JDL offers a variety of different features that make it a great platform for streamlining the development of modern web applications. Below, are some of the key features the platform provides:

- **Entity Declaration:** With JDL, developers can define core aspects of their application by splitting these aspects into entities, which allows developers to to concisely specify the attributes and relations of these key aspects of the program. Essentially, this makes the development of applications more user friendly, in the sense that developers are able to perhaps focus on one core entity at a time, where they can figure out all the attributes they would like to apply to a single entity and also the relations of that entity to another.
- **Validation Constraints:** Developers can easily define and specify any attribute constraints relating to a given entity. Common constraints are; string length limits(min/max), required fields, regular expression(regex) patterns.
- **Relationship Management:** JDL, offers user-friendly methods of defining relations between entities, e.g. one-to one, one-to-many and many-to-one associations. The benefit of using JDL in relationship management, is that once these associations are defined JDL is able to generate the necessary code and database configurations to implement these relationships.
- **JHipster Integration:** The integration of JDL within JHipster, offers a great tool for generating full stack applications. As mentioned previously, while JDL can be used to define the domain model of an application, through its integration with JHipster, JHipster can then generate the necessary code, including entity classes, database configurations, REST APIs, and AngularJS or React components.

Integration into the eSports Organiser

In relation to the eSports Organiser, using JDL will allow you to define key entities (e.g. teams) along with their attributes and relationships which altogether represent the core functionalities of the application. This essentially enables developers to parse the JDL file and generate the necessary code representing the main entities, as well as having the ability to generate front-end components to provide interaction with the features of the application (create/join team for example). A few examples of how JDL could be used:

Entities

Entity creation:

```
Entity User {
   username String required,
   password String required minlength(8)
}
```

An entity is created for a user containing attributes *username* and *password* which both hold validation constraints of requirement, in addition to the password having a minimum length constraint of 8.

Relationships

Relationship creation:

```
Relationship ManyToOne {
    User to Team
}
```

This relationship defines the association between entities, user and *team* acknowledging the fact that teams can have many users, however users can only have one team, therefore the relation **ManyToOne.**

Data Transfer Objects

Data Transfer Object definition:

```
Dto UserDTO {
    username,
    email
}
```

Data Transfer Objects (DTO) encapsulates entities into objects so that data can be transferred from these entities across the front-end and back-end. Here we see a DTO being created for the user entity, it essentially acts a security measure for this particular entity, as instead of transferring the entire user object, with potential sensitive information, a dto is created containing only the username and email. This ensures that confidential information is not exposed to the front-end.

Databases

JDL can also be viewed as a database related technology. Through entity creation, these entities directly map to database tables, where each entity attribute corresponds to a column or field in the database. Additionally, through relationship definition (for example one-to-one etc), these relationships can be directly reflected into a database model, where the database would interpret these relationships and present them through foreign key constraints, if using relational databases such PostgreSQL. With this being said, JHipster offers tools that allow you to parse a JDL file to generate database migration scripts, which provide instructions for creating the database table with the specified data from the JDL file.