The JDL

Creating and managing your applications with the JDL

Who I am

Mathieu Abou-Aichi (MathieuAA on GitHub)

Co-creator of JHipster UML & JHipster Core (the JDL)

Maintainer of the two projects

What the JDL is

A JHipster-exclusive domain-specific language

Used to generate entities... and applications

Doesn’t make your coffee yet

What I’m going to talk about

Entity & Application generation

What I’m going to talk about

Entity & Application generation

Managing JDL files

What I’m going to talk about

Entity & Application generation

Managing JDL files (Spoiler, new feature inside)

What I’m going to talk about

Entity & Application generation

Managing JDL files (Spoiler, new feature inside)

Future features

Entity & Application generation

Entity generation

```
MAX_LENGTH=42
entity Conference {
  name String required maxlength(MAX_LENGTH)
}
entity Attendee {
  name String required
  email String required
}
relationship ManyToMany {
  Event to Attendee
}
dto * with mapstruct
service * with serviceClass
```

Handled:

- entities
- relationships
- options
- enums
- constants (only in the JDL)

Application generation

```
application {
  config {
    baseName MySuperApp,
    applicationType microservice,
    serverPort 8090,
    ...
  }
}
```

Supports every application option available

Converted to a **.yo-rc.json** file

Used by JHipster to generate your app

Behind the scenes

Input: 1 or more JDL files

A simple command: *jhipster import-jdl jdl_1.jdl ... jdl_n.jdl*

A parser written with **PegJS** transforms it into a readable format

JHipster Core exports the entity & application files

The JDL import sub-generator imports everything by using the other generators

Managing JDL files

Managing JDL files

WHY?!

JDL Linter

Parses your JDL and checks everything's clean

JDL Linter

Parses your JDL and checks everything's clean

Some rules: grouped relationships, useless characters, clean options

JDL Linter

Parses your JDL and checks everything's clean

Some rules: grouped relationships, useless characters, clean options

Configurable rules

JDL Linter

Parses your JDL and checks everything's clean

Some rules: grouped relationships, useless characters, clean options

Configurable rules

((Not finished yet))

Best practices

Use Antonio's "side-by-side" approach

Keep your JDL files clean / use the linter

Use JDL Studio or JHipster IDE

Future features

Future features

Integration of **chevrotain**

Future features

Integration of **chevrotain**

JDL support for other sub-generators

Future features

Integration of **chevrotain**

JDL support for other sub-generators

Annotations for the JDL

Thank you!