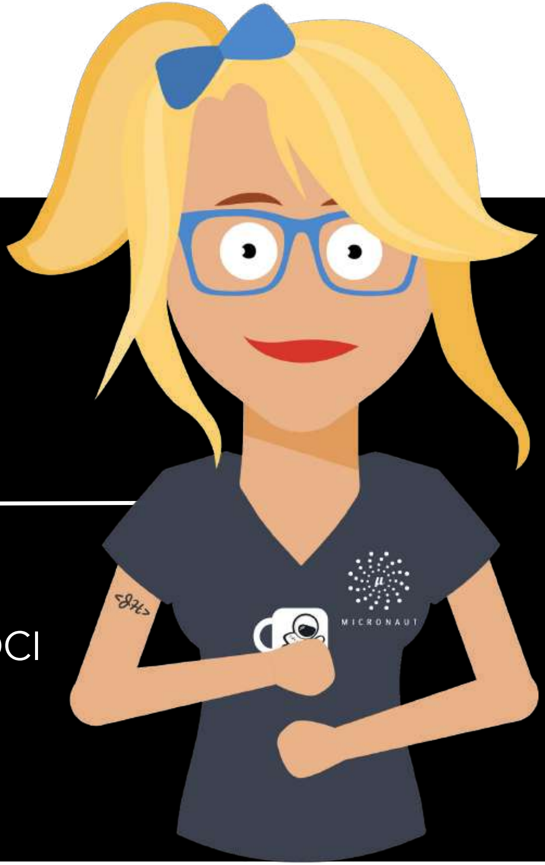
OBJECT COMPUTING
Home to the MICRONAUT® framework

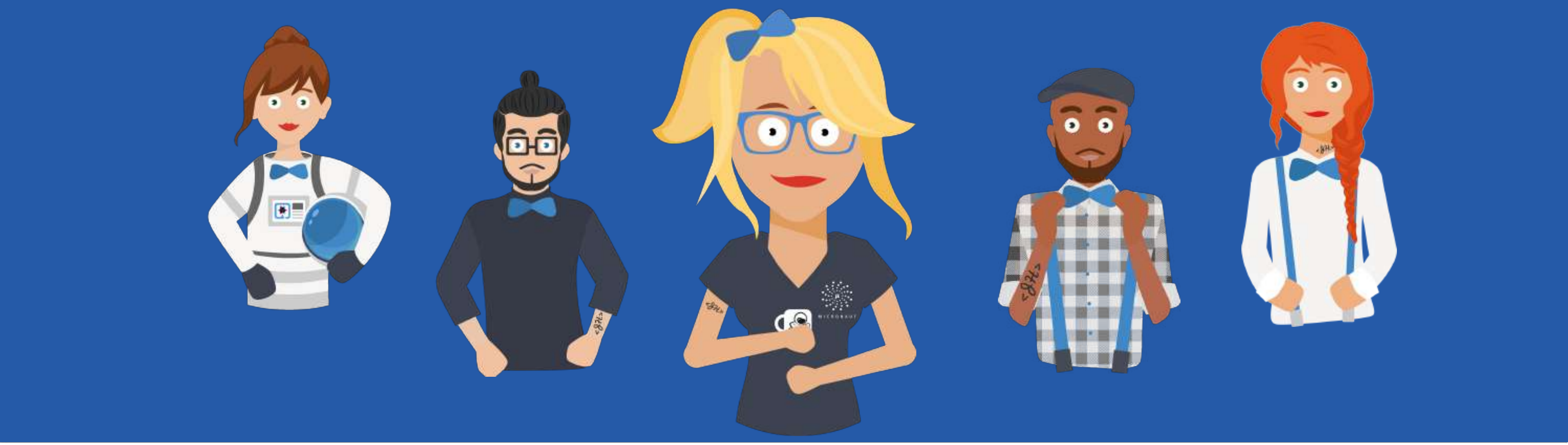
Getting Started with the JHipster Micronaut Blueprint

Frederik Hahne
JHipster Team Member
@atomfrede

Jason Schindler
2GM Team Manager & Partner @ OCI
@JasonTypesCodes



Micronaut Blueprint for JHipster v1.0 Released!



JHipster is is a development platform to quickly generate, develop, & deploy modern web applications & microservice architectures.

A high-performance robust server-side stack with excellent test coverage

A sleek, modern, mobile-first UI with Angular, React, or Vue + Bootstrap for CSS

A powerful workflow to build your application with Webpack and Maven or Gradle

Infrastructure as code so you can quickly deploy to the cloud

JHipster in Numbers

- 18K+ Github Stars
- 600+ Contributors on the main generator
- 50K registered users on start.jhipster.tech
- 40K+ weekly download via npmjs.com
- 100K annual budget from individual and institutional sponsors
- Open Source under Apache License
- 51% JavaScript, 20% TypeScript, 18% Java

JHipster Overview

- Platform to quickly generate, develop, & deploy modern web applications & microservice architectures.
- Started in 2013 as a bootstrapping generator to create Spring Boot + AngularJS applications
- Today creating production ready application, data entities, unit-, integration-, e2e-tests, deployments and ci-cd configurations
- Extensibility via modules or blueprints
- Supporting wide range of technologies from the JVM and non-JVM ecosystem
  - E.g. node.js or .net as backend stack
  - SQL databases, noSQL databases, different test frameworks

Extending JHipster

- There have been always limited capabilities to extend or adapt JHipster features (e.g. adding new dependencies, small adaptations to the build process)
- Larger changes have not been possible without adding it or changing the JHipster core which is hard to maintain
- Blueprints are JHipster's solution to that problem
- Blueprints are composed with the respective core blueprint
- JHipster uses blueprints internally for nearly all parts already
- Use the configuration options/framework of JHipster but create totally different or modified set of files
- Enables wide range of possibilities as you have full control what to change and how
- Blueprints can be again combined (e.g. using Micronaut + vue.js)

MICRONAUT®

The Micronaut® framework is a modern, Open Source, JVM-based, full-stack framework for building modular, easily testable microservice and serverless applications.

Monumental Leap in Startup Time

Blazing-Fast Throughput

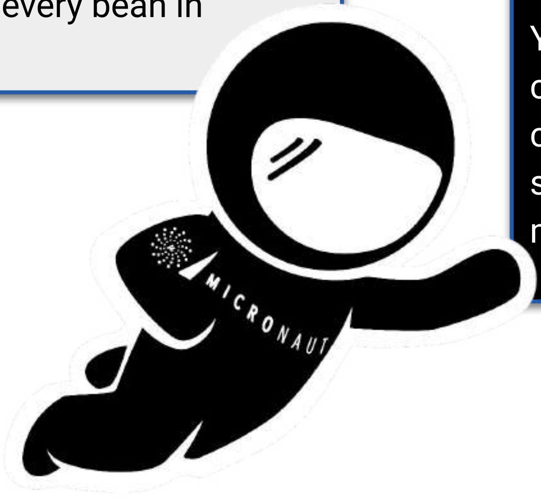
Minimal Memory Footprint

Cloud-Native Architecture

REIMAGINE STARTUP TIME AND MEMORY CONSUMPTION

The Old Way

When building applications with reflection-based IoC frameworks, the framework loads and caches reflection data for every single field, method, and constructor for every bean in the application context.



The Micronaut® Way

Your application startup time and memory consumption aren't bound to the size of your codebase, resulting in a monumental leap in startup time, blazing-fast throughput, and a minimal memory footprint.

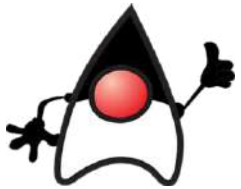
TECHNICAL OVERVIEW

- Robust full-stack framework for building microservices and serverless applications
- Smooth learning curve makes it as easy for JVM developers to adopt
- Easily spin up servers and clients in your unit tests, and run them instantaneously
- Provides a simple compile-time aspect-oriented programming API that does not use reflection
- Supports Java, Groovy, Kotlin (Scala on the roadmap)
- Supports any framework that implements reactive streams, including RxJava, and Reactor
- Native-Image compatible

FLEXIBILITY

POLYGLOT FRAMEWORK

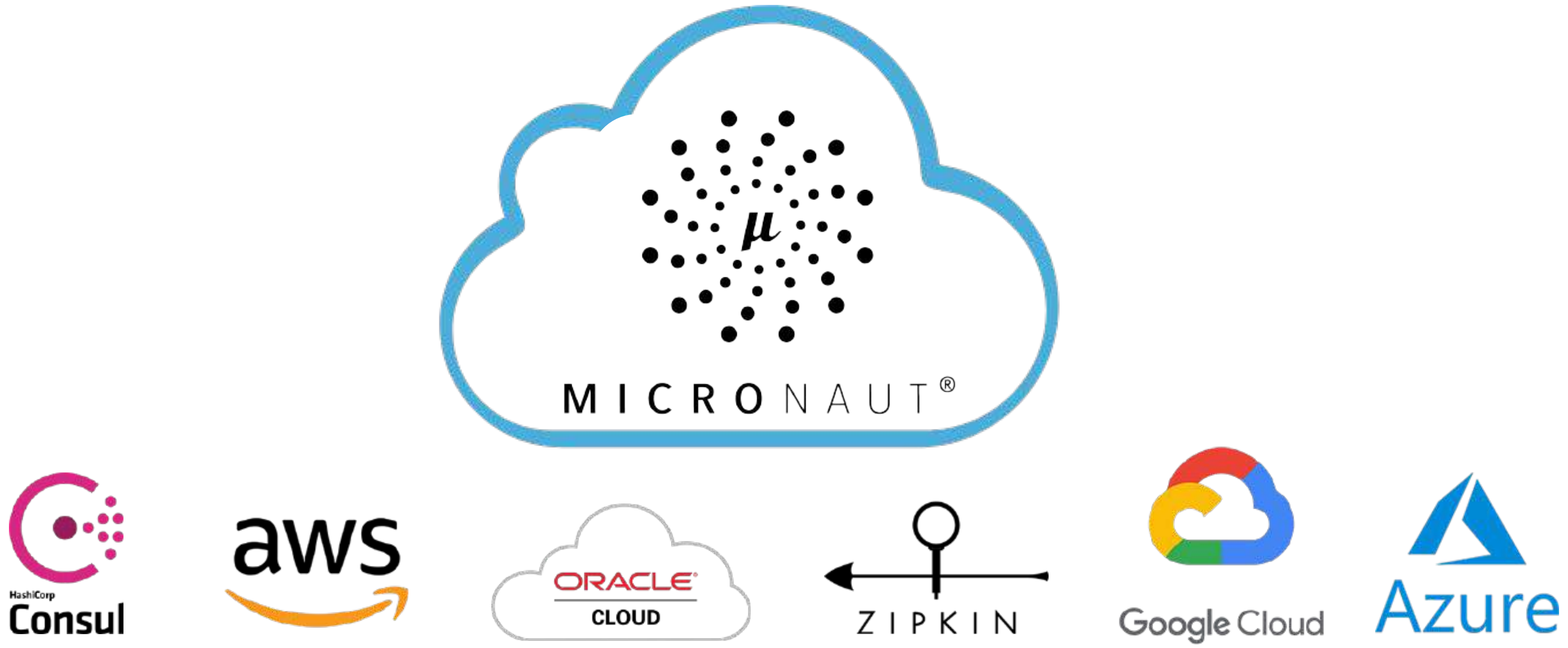
- Micronaut apps start up in tens of milliseconds with GraalVM!
- Micronaut features a dependency injection and aspect-oriented programming runtime that uses no reflection. This makes it easier for Micronaut applications to run on GraalVM.
- Micronaut 1 and 2 are each tested with Graal native images.



- We have no restrictions on the distribution or vendor of the JDK and have no knowledge of any incompatibilities between versions 8 and 13 for v1 and 8 and 14 for v2.
- Micronaut 1 is tested with the Zulu distribution of OpenJDK 8, 11, and 13.
- Micronaut 2 is tested with Zulu distribution of OpenJDK 8, 11, 14, and the Amazon Corretto distribution of OpenJDK 8 and 11.

NATIVELY CLOUD NATIVE

The Micronaut framework's cloud support is built right in, including support for common discovery services, distributed tracing tools, and cloud runtimes.



MHipster 1.0 Features

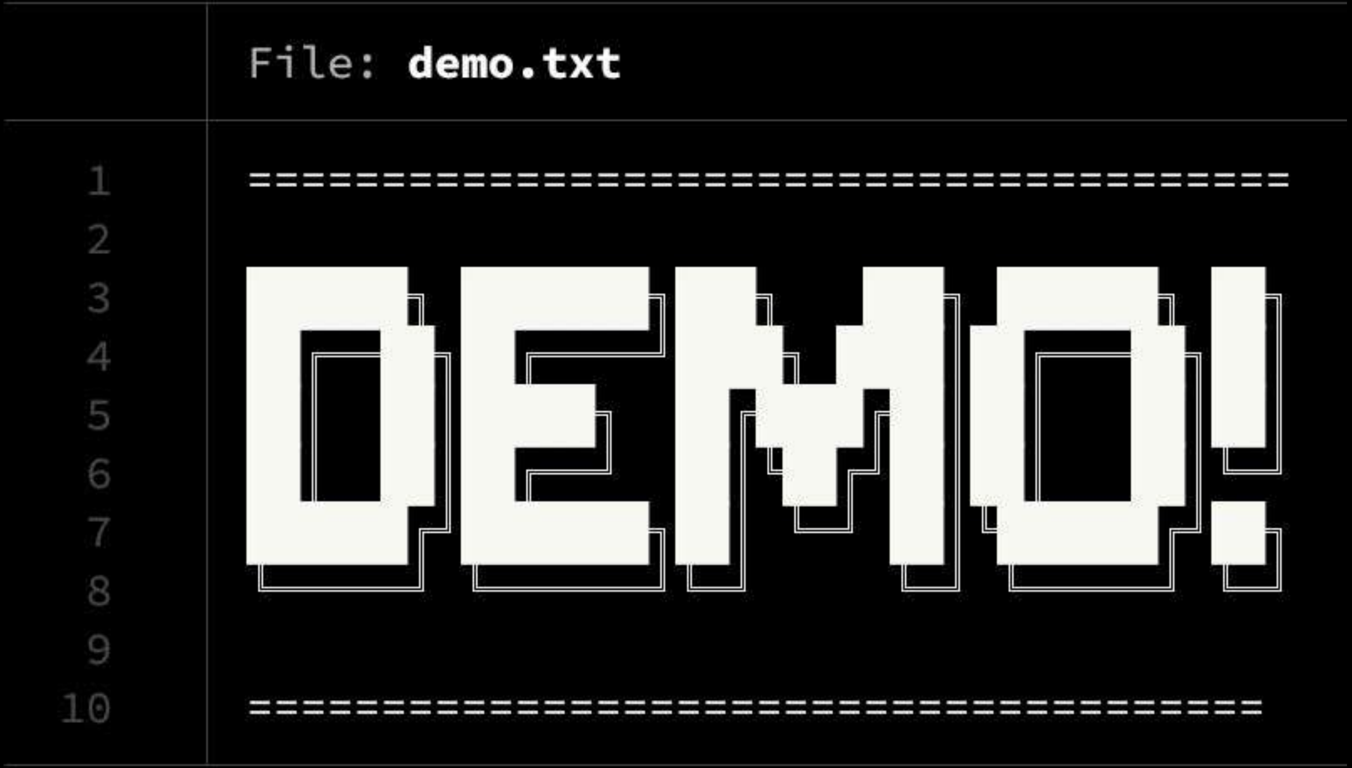
- Monolith / Microservice Applications
- JWT or OAuth 2.0 Authentication
- Gradle / Maven Builds
- SQL Database Support
  - MySQL
  - MariaDB
  - PostgreSQL
  - H2
- Caching
  - Ehcache
  - Caffeine
  - Redis
- Angular / React Client Application
- Protractor Tests
- Heroku Deployment

Getting Started

```
$ npm install -g generator-jhipster@6.10.5 generator-jhipster-micronaut@1.0.0
```

```
$ mkdir hello-mhipster && cd hello-mhipster

$ mhipster
```

Creating an Application with the CLI

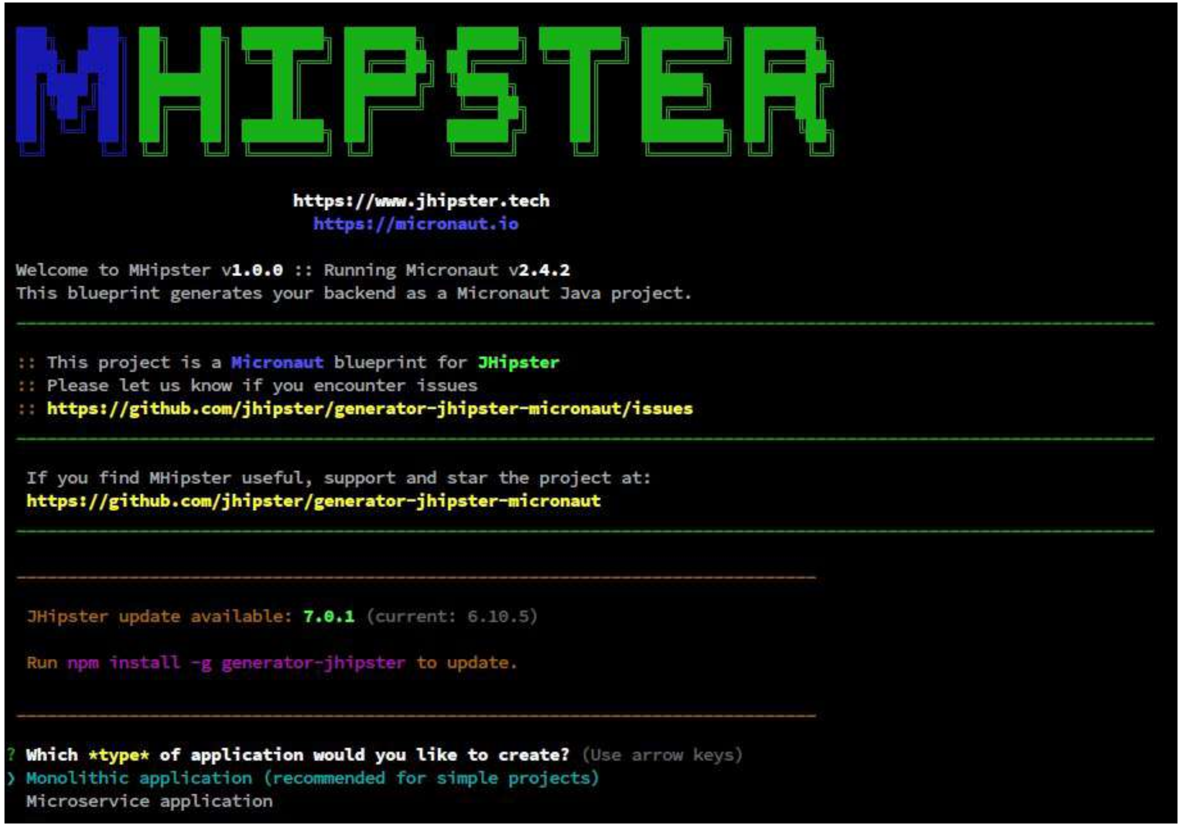


File: demo.txt

```
=======================================
```

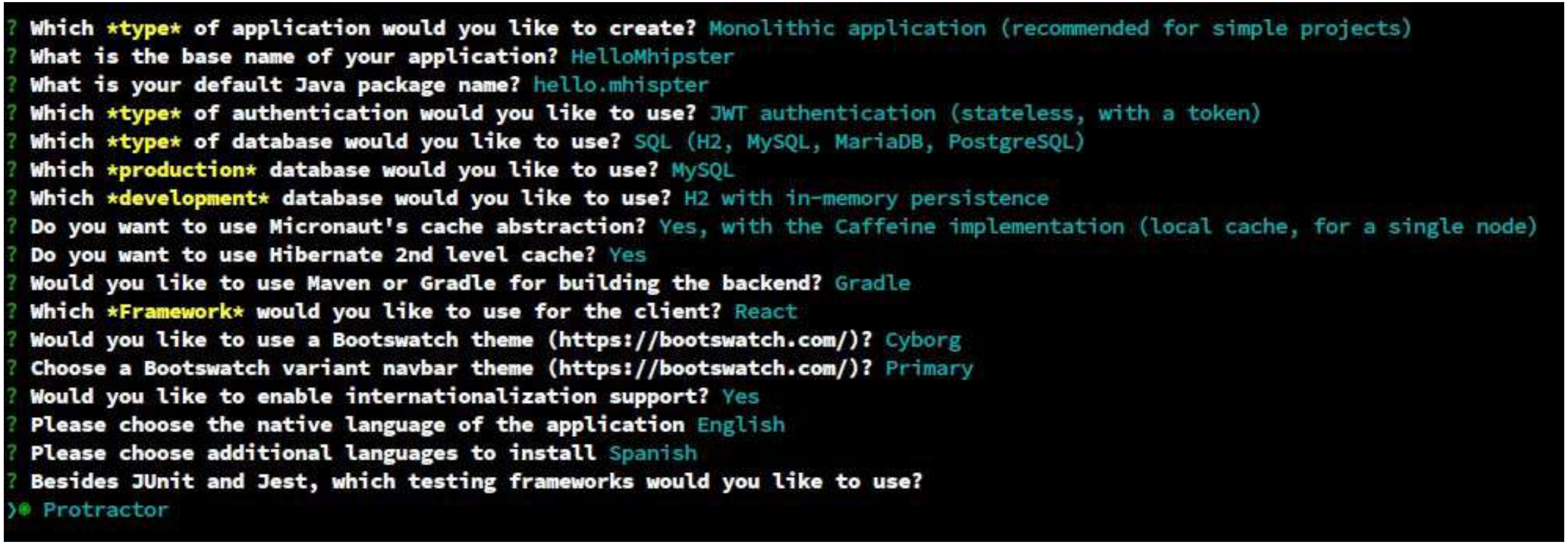
```
======================================
```

Creating an Application with the CLI



```
MHIPSTER

                         https://www.jhipster.tech
                          https://micronaut.io

Welcome to MHipster v1.0.0 :: Running Micronaut v2.4.2
This blueprint generates your backend as a Micronaut Java project.
____________________________________________________________________________________________________

:: This project is a Micronaut blueprint for JHipster
:: Please let us know if you encounter issues
:: https://github.com/jhipster/generator-jhipster-micronaut/issues
____________________________________________________________________________________________________

 If you find MHipster useful, support and star the project at:
 https://github.com/jhipster/generator-jhipster-micronaut
____________________________________________________________________________________________________


______________________________________________________________________________

 JHipster update available: 7.0.1 (current: 6.10.5)

 Run npm install -g generator-jhipster to update.

______________________________________________________________________________

? Which *type* of application would you like to create? (Use arrow keys)
> Monolithic application (recommended for simple projects)
  Microservice application
```

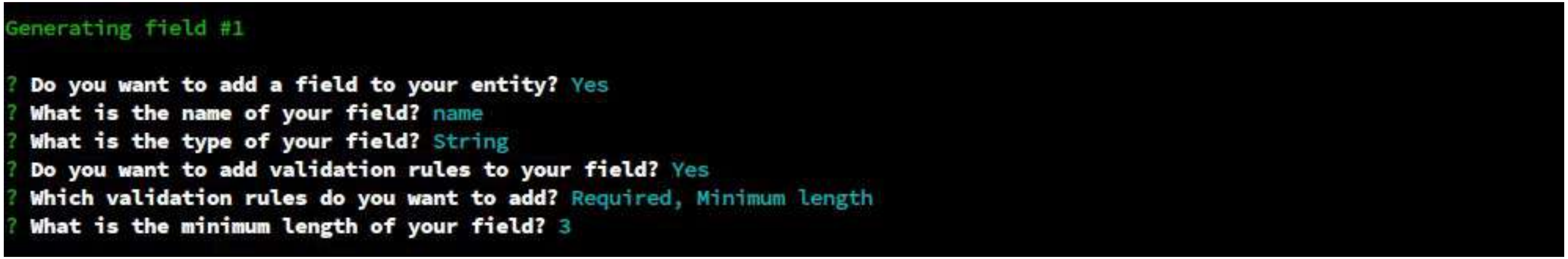
Creating an Application with the CLI



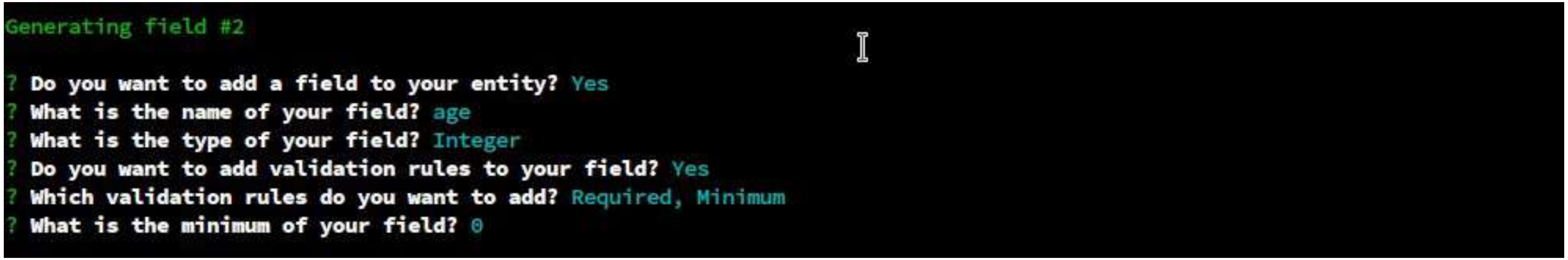
```
? Which *type* of application would you like to create? Monolithic application (recommended for simple projects)
? What is the base name of your application? HelloMhipster
? What is your default Java package name? hello.mhispter
? Which *type* of authentication would you like to use? JWT authentication (stateless, with a token)
? Which *type* of database would you like to use? SQL (H2, MySQL, MariaDB, PostgreSQL)
? Which *production* database would you like to use? MySQL
? Which *development* database would you like to use? H2 with in-memory persistence
? Do you want to use Micronaut's cache abstraction? Yes, with the Caffeine implementation (local cache, for a single node)
? Do you want to use Hibernate 2nd level cache? Yes
? Would you like to use Maven or Gradle for building the backend? Gradle
? Which *Framework* would you like to use for the client? React
? Would you like to use a Bootswatch theme (https://bootswatch.com/)? Cyborg
? Choose a Bootswatch variant navbar theme (https://bootswatch.com/)? Primary
? Would you like to enable internationalization support? Yes
? Please choose the native language of the application English
? Please choose additional languages to install Spanish
? Besides JUnit and Jest, which testing frameworks would you like to use?
>◉ Protractor
```

Creating an Application with the CLI

```
$ mhipster entity Fish
```

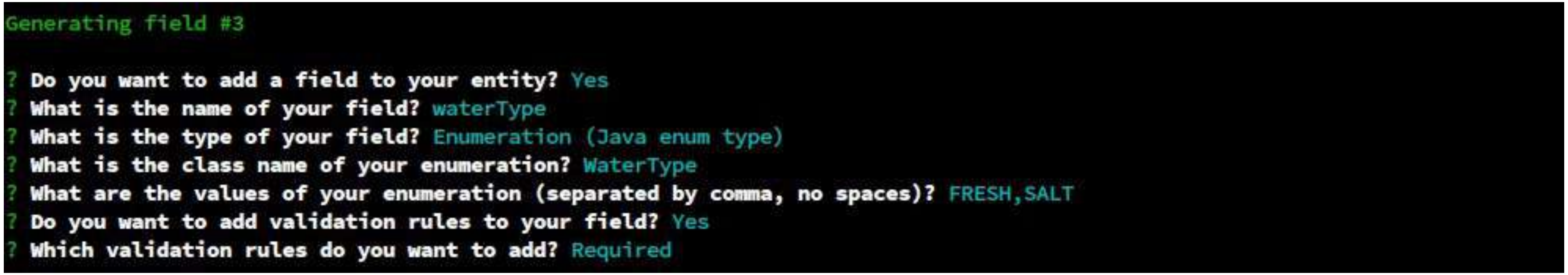


```
Generating field #1

? Do you want to add a field to your entity? Yes
? What is the name of your field? name
? What is the type of your field? String
? Do you want to add validation rules to your field? Yes
? Which validation rules do you want to add? Required, Minimum length
? What is the minimum length of your field? 3
```



```
Generating field #2

? Do you want to add a field to your entity? Yes
? What is the name of your field? age
? What is the type of your field? Integer
? Do you want to add validation rules to your field? Yes
? Which validation rules do you want to add? Required, Minimum
? What is the minimum of your field? 0
```

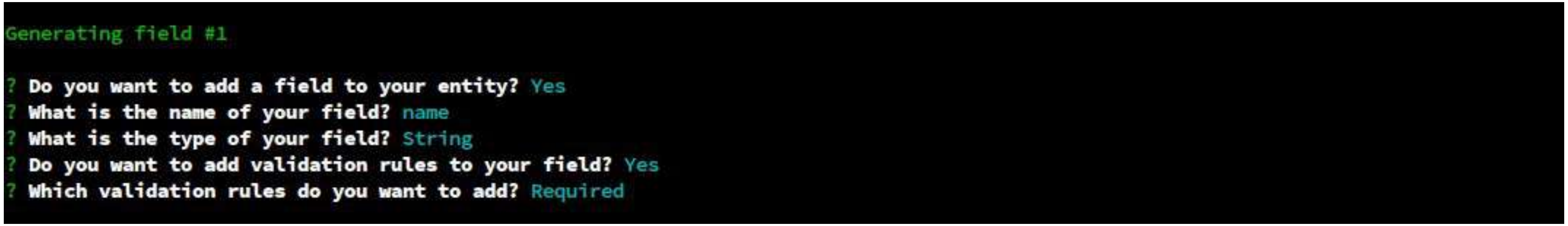
Creating an Application with the CLI



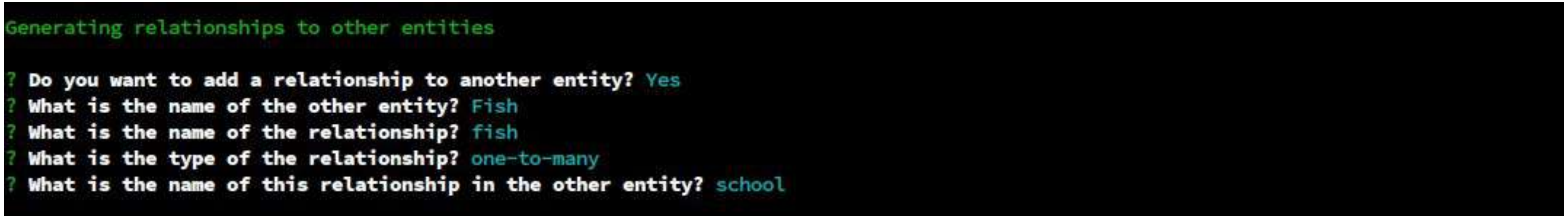
```
Generating field #3

? Do you want to add a field to your entity? Yes
? What is the name of your field? waterType
? What is the type of your field? Enumeration (Java enum type)
? What is the class name of your enumeration? WaterType
? What are the values of your enumeration (separated by comma, no spaces)? FRESH,SALT
? Do you want to add validation rules to your field? Yes
? Which validation rules do you want to add? Required
```

Creating an Application with the CLI

```
$ mhipster entity School
```



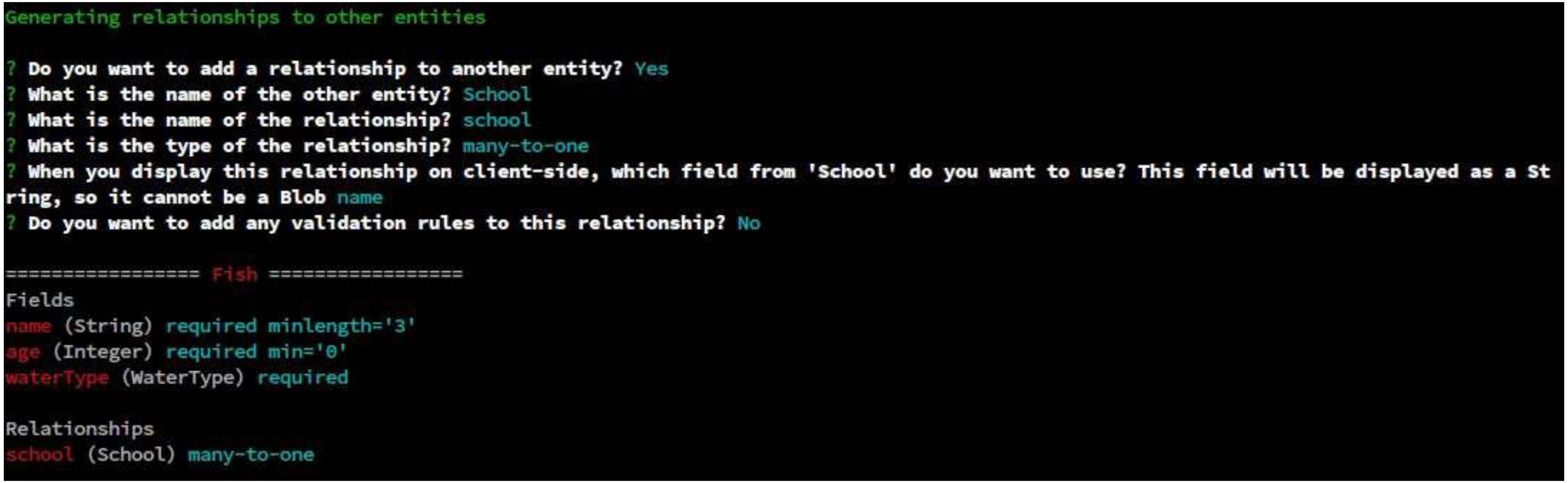
```
Generating field #1

? Do you want to add a field to your entity? Yes
? What is the name of your field? name
? What is the type of your field? String
? Do you want to add validation rules to your field? Yes
? Which validation rules do you want to add? Required
```



```
Generating relationships to other entities

? Do you want to add a relationship to another entity? Yes
? What is the name of the other entity? Fish
? What is the name of the relationship? fish
? What is the type of the relationship? one-to-many
? What is the name of this relationship in the other entity? school
```

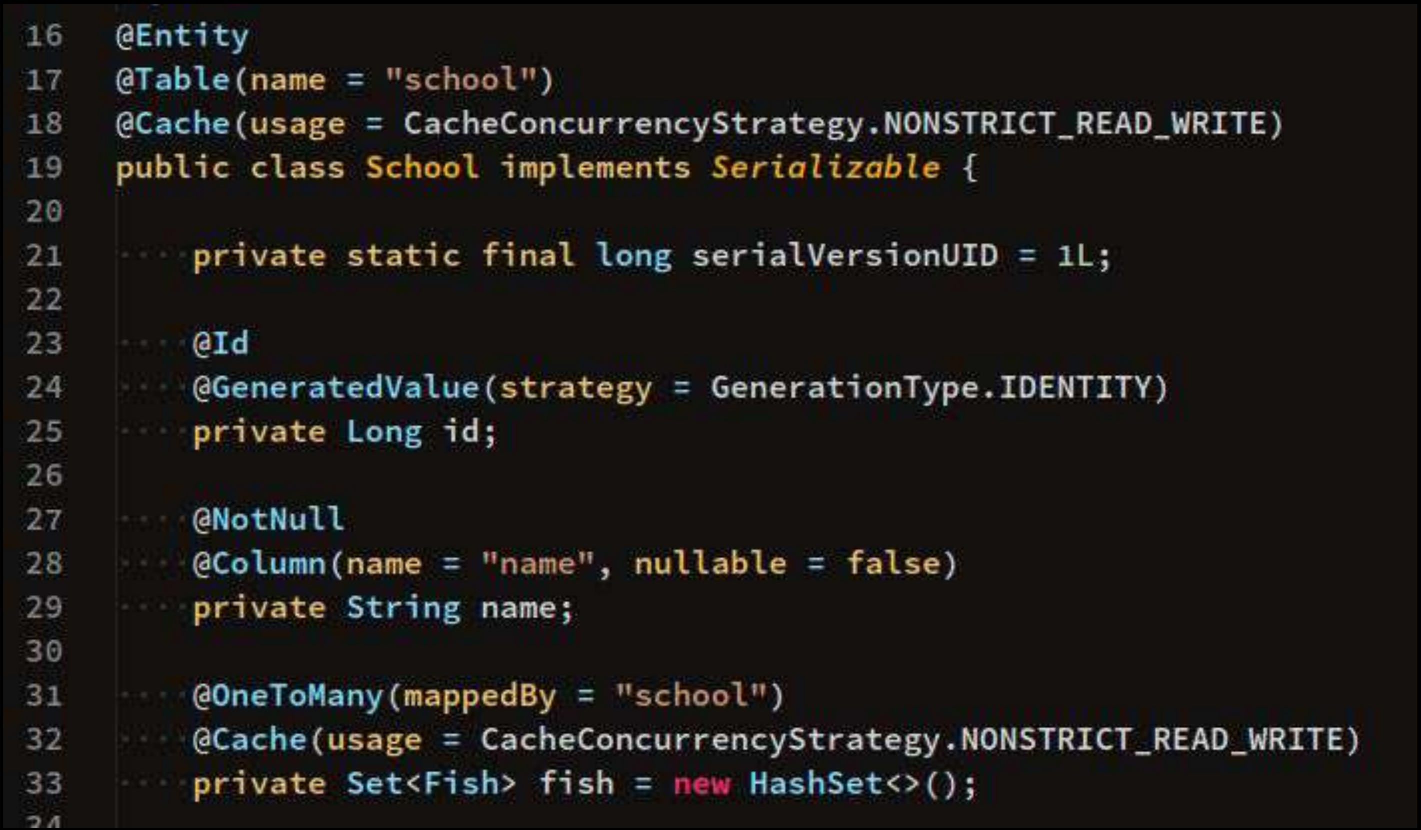
Creating an Application with the CLI

```
$ mhipster entity Fish
```



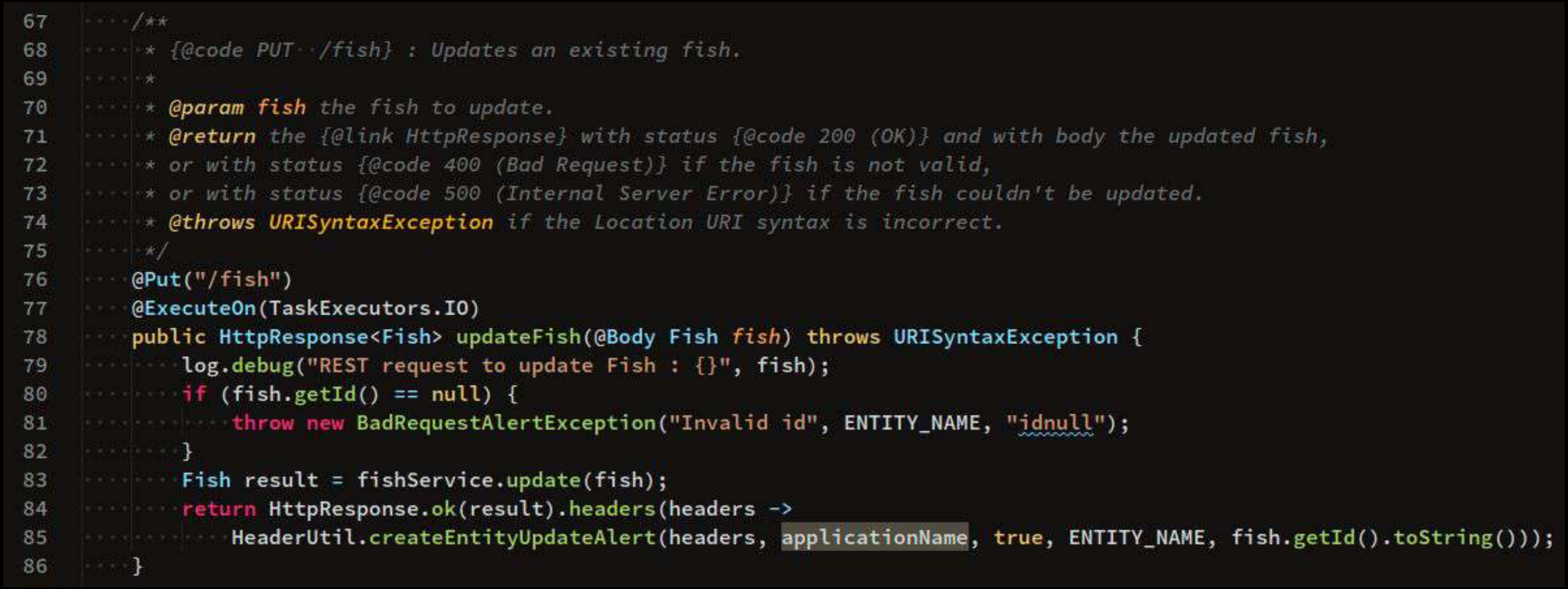
```
Generating relationships to other entities

? Do you want to add a relationship to another entity? Yes
? What is the name of the other entity? School
? What is the name of the relationship? school
? What is the type of the relationship? many-to-one
? When you display this relationship on client-side, which field from 'School' do you want to use? This field will be displayed as a St
ring, so it cannot be a Blob name
? Do you want to add any validation rules to this relationship? No

================= Fish =================
Fields
name (String) required minlength='3'
age (Integer) required min='0'
waterType (WaterType) required

Relationships
school (School) many-to-one
```

Creating an Application with the CLI



```java
@Entity
@Table(name = "school")
@Cache(usage = CacheConcurrencyStrategy.NONSTRICT_READ_WRITE)
public class School implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull
    @Column(name = "name", nullable = false)
    private String name;

    @OneToMany(mappedBy = "school")
    @Cache(usage = CacheConcurrencyStrategy.NONSTRICT_READ_WRITE)
    private Set<Fish> fish = new HashSet<>();
```

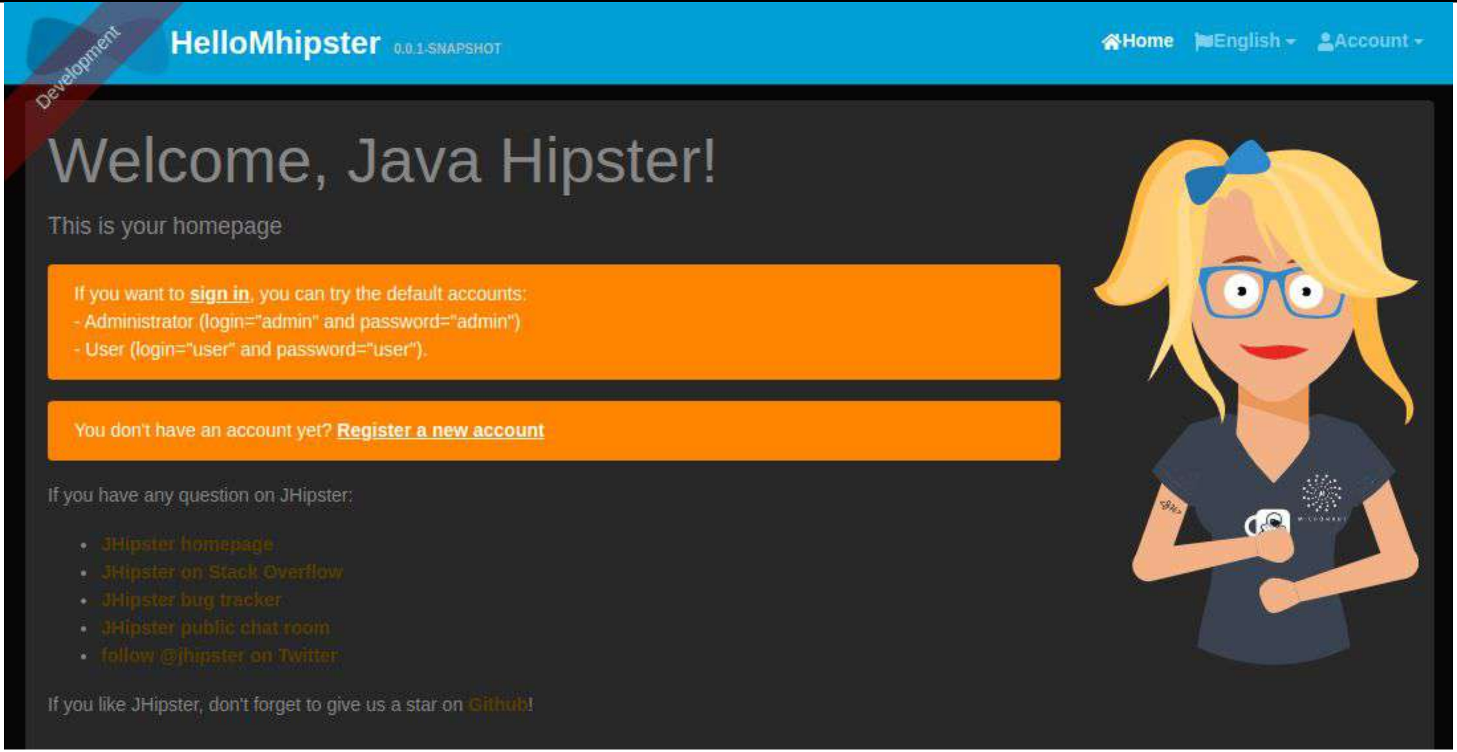
Creating an Application with the CLI



```java
    /**
     * {@code PUT  /fish} : Updates an existing fish.
     *
     * @param fish the fish to update.
     * @return the {@link HttpResponse} with status {@code 200 (OK)} and with body the updated fish,
     * or with status {@code 400 (Bad Request)} if the fish is not valid,
     * or with status {@code 500 (Internal Server Error)} if the fish couldn't be updated.
     * @throws URISyntaxException if the Location URI syntax is incorrect.
     */
    @Put("/fish")
    @ExecuteOn(TaskExecutors.IO)
    public HttpResponse<Fish> updateFish(@Body Fish fish) throws URISyntaxException {
        log.debug("REST request to update Fish : {}", fish);
        if (fish.getId() == null) {
            throw new BadRequestAlertException("Invalid id", ENTITY_NAME, "idnull");
        }
        Fish result = fishService.update(fish);
        return HttpResponse.ok(result).headers(headers ->
            HeaderUtil.createEntityUpdateAlert(headers, applicationName, true, ENTITY_NAME, fish.getId().toString()));
    }
```

Creating an Application with the CLI

```
$ ./gradlew
```



Welcome, Java Hipster!

This is your homepage

If you want to **sign in**, you can try the default accounts:
- Administrator (login="admin" and password="admin")
- User (login="user" and password="user").

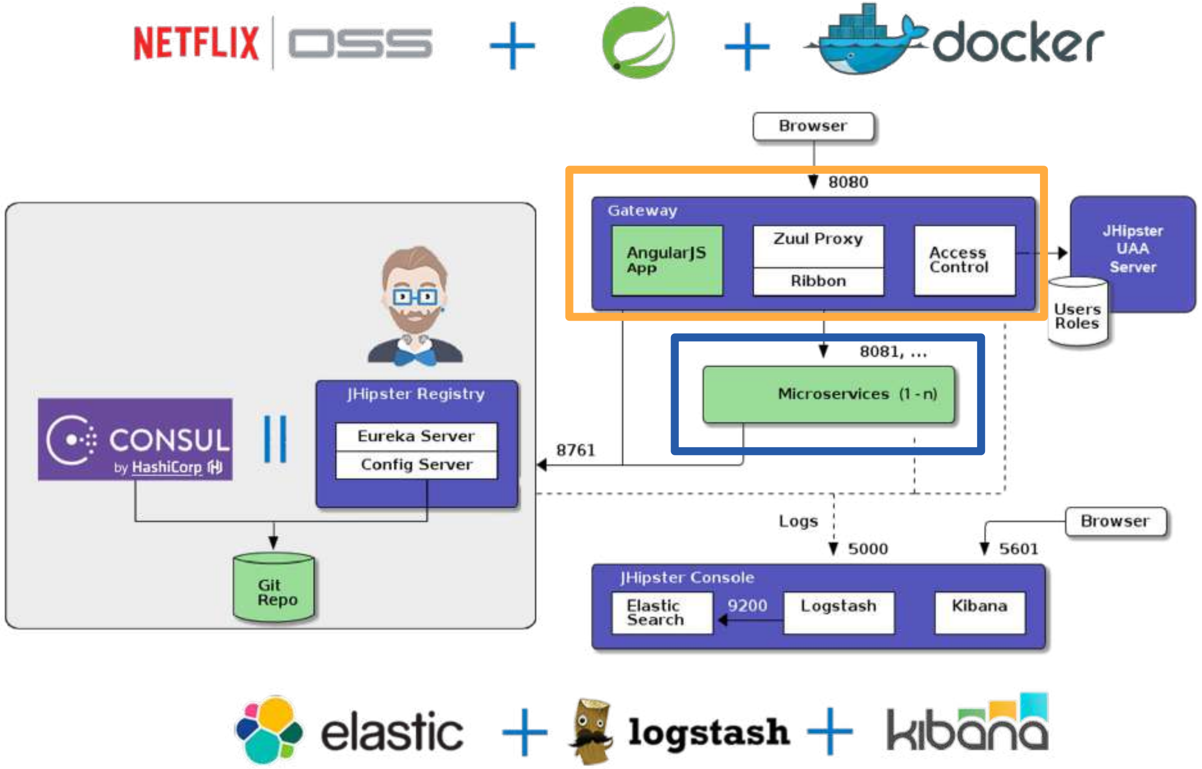
You don't have an account yet? **Register a new account**

If you have any question on JHipster:

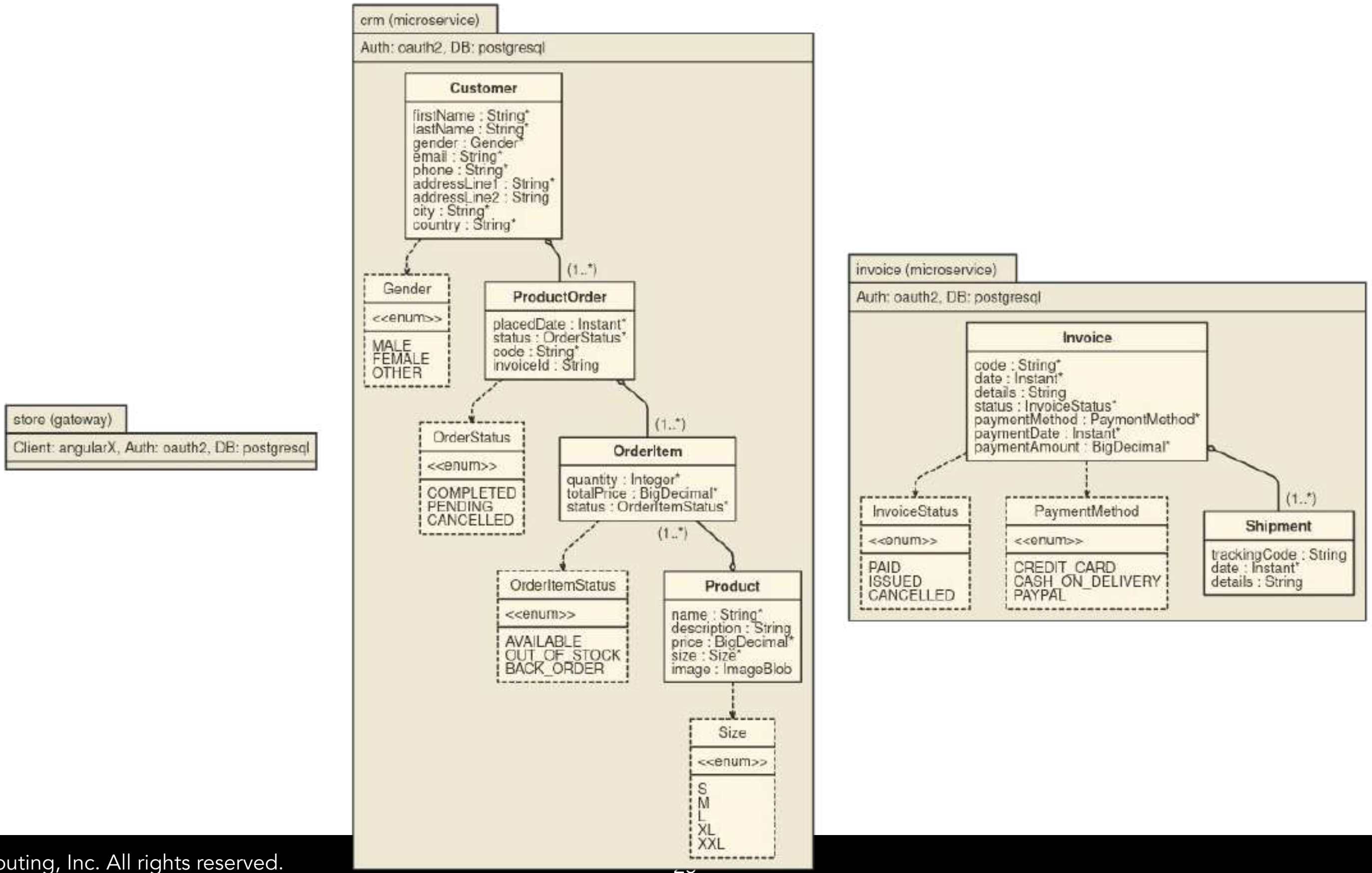
- JHipster homepage
- JHipster on Stack Overflow
- JHipster bug tracker
- JHipster public chat room
- follow @jhipster on Twitter

If you like JHipster, don't forget to give us a star on Github!

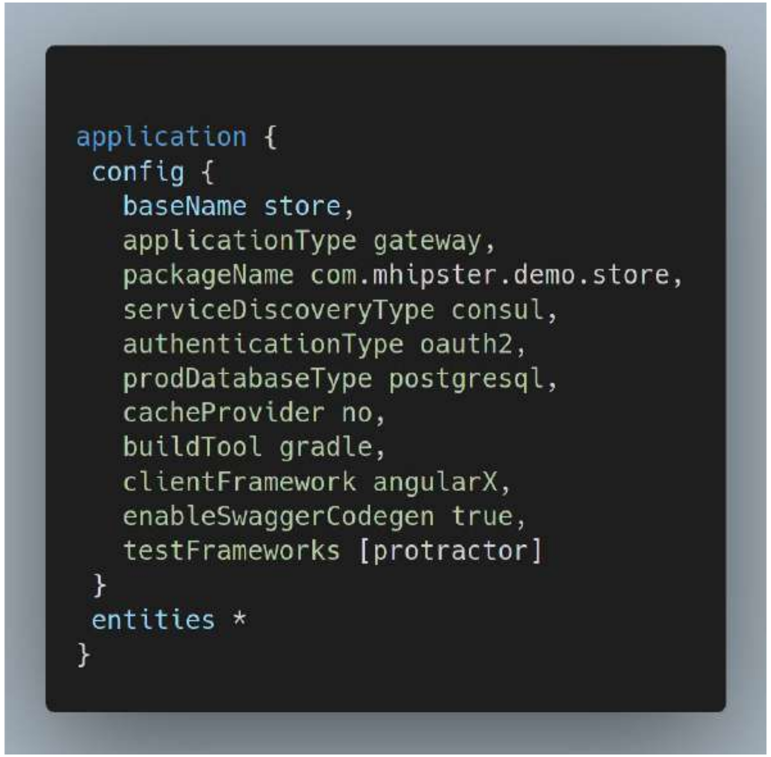
Example microservice application walkthrough (JDL)



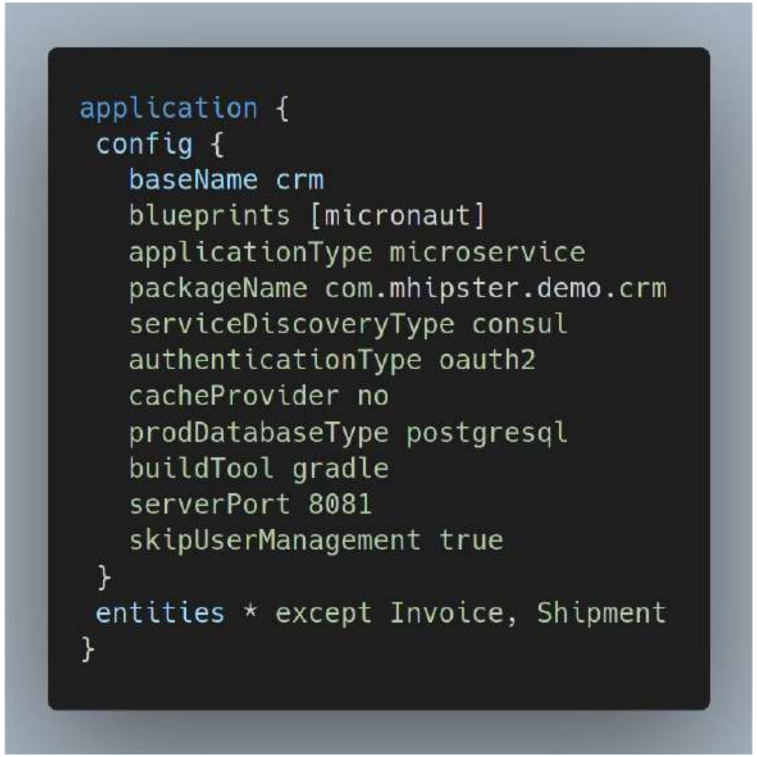
Example microservice application walkthrough (JDL)



Example microservice application walkthrough (JDL)



```
application {
 config {
   baseName store,
   applicationType gateway,
   packageName com.mhipster.demo.store,
   serviceDiscoveryType consul,
   authenticationType oauth2,
   prodDatabaseType postgresql,
   cacheProvider no,
   buildTool gradle,
   clientFramework angularX,
   enableSwaggerCodegen true,
   testFrameworks [protractor]
 }
 entities *
}
```



```
application {
 config {
   baseName crm
   blueprints [micronaut]
   applicationType microservice
   packageName com.mhipster.demo.crm
   serviceDiscoveryType consul
   authenticationType oauth2
   cacheProvider no
   prodDatabaseType postgresql
   buildTool gradle
   serverPort 8081
   skipUserManagement true
 }
 entities * except Invoice, Shipment
}
```

Example microservice application walkthrough (JDL)

```
$ jhipster import-jdl jhipster.jdl
$ docker-compose -f docker-compose/consul.yml up -d
$ docker-compose -f docker-compose/keycloak.yml up -d
$ store|crm|invoice >> ./gradlew
---
$ store >> ./gradlew jibDockerBuild
$ crm|invoice >> ./gradlew dockerBuild
$ docker-compose -f docker-compose/docker-compose.yml up -d
```

What’s next?

- JHipster 7 upgrade
- Support Micronaut as gateway application
- Support reactive applications
- Support native images
- Fix existing bugs and streamline user experience
- Better documentation

Interested in Contributing?

- Reporting or validating issues
- Requesting features
- Fixing documentation
- Fixing bugs
- Implementing features or enhancements

Interested in Contributing?

Best place to start is on the project GitHub page:

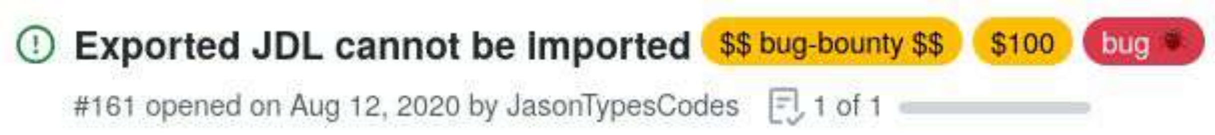
- https://github.com/jhipster/generator-jhipster-micronaut

Make sure you read Code of Conduct and Contributing documentation

- https://github.com/jhipster/generator-jhipster-micronaut/blob/main/CODE_OF_CONDUCT.md
- https://github.com/jhipster/generator-jhipster-micronaut/blob/main/CONTRIBUTING.md

Review current Issues and lookout for Bug Bounties!

- https://www.jhipster.tech/bug-bounties/#-bug-bounties


Exported JDL cannot be imported $$ bug-bounty $$ $100 bug

#161 opened on Aug 12, 2020 by JasonTypesCodes 1 of 1

Sponsorship Opportunities

- JHipster through OpenCollective:
    - https://opencollective.com/generator-jhipster#sponsor
- JHipster Association:
    - French Non-Profit Organization
    - Owning the copyright of source code
    - Creating a Technology Advisory Board to ensure the advancement of JHipster
    - At the moment 16 members from 6 countries
    - Organize the annual JHipster conference
- Micronaut through Micronaut Foundation:
    - https://micronaut.io/foundation/

MICRONAUT® FOUNDATION

- Micronaut Foundation is a not-for-profit organization that exists to support and collectively lead the open source Micronaut® project
- Micronaut Foundation is supported by a Technology Advisory Board that ensures the Framework continues to reflect and serve its diverse and growing user community
- Micronaut Foundation serves to:
    - Ensure technical innovation and advancement of the Micronaut framework as a free and open public use software development framework for a growing global community
    - Evangelize and promote the Micronaut framework as a leading technology in the JVM space
    - Build and support an ecosystem of complementary documentation, functionality, and services.
- For more information on becoming a contributing member and for organizations interested in nominating a representative to the Micronaut Foundation Technical Advisory Board, please contact us at foundation@micronaut.io

COMMUNITY RESOURCES

- https://github.com/jhipster/generator-jhipster-micronaut
- https://www.jhipster.tech/
- https://micronaut.io/
- https://www.jhipster.tech/jdl-studio/
- https://opencollective.com/generator-jhipster#sponsor
- https://micronaut.io/foundation/
- https://gitter.im/micronautfw
- https://objectcomputing.com/products/micronaut/consulting-support
- https://micronaut.io/launch
- https://micronaut.io/guides
- https://micronaut.io/docs/

Demo Code From Today’s Webinar

- https://github.com/oci-labs/mhipster-webinar-demos

THANK YOU!

LET'S CONNECT

@micronautfw

info@micronaut.io