OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Build a Better API with GORM & GraphQL

Zachary Klein, Senior Software Engineer
STL JUG
October 10 2019

© 2019, Object Computing, Inc. (OCI). All rights reserved. No part of these notes may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior, written permission of Object Computing, Inc. (OCI)

OUR GLOBAL FOOTPRINT

170+ ENGINEERS
MS/PHD
Advanced Security Clearance

19 AVERAGE YEARS
Engineering Experience

14 AVERAGE YEARS
OCI Tenure

© 2019, Object Computing, Inc. (OCI). All rights reserved.

AREAS OF EXPERTISE

We deliver mission-critical software solutions that accelerate innovation within your organization and stand up to the evolving demands of your business.

INDUSTRIAL IoT

We equip industrial environments with seamless connectivity and real-time analytics that reduce operating costs and deliver on customer demands.

MACHINE LEARNING

We can modernize your legacy applications and enable scalable, integrated AI capabilities tailored to your unique sets of data.

BLOCKCHAIN SOLUTIONS

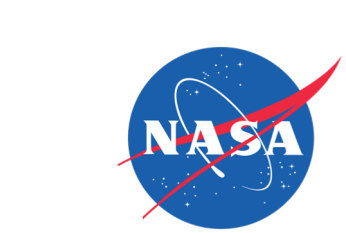
We are at the forefront of blockchain technology, and we have practical, real-world experience with its implementation.

CLOUD ENGINEERING

We combine software engineering expertise with cloud-native architecture to accelerate innovation within your organization.

© 2019, Object Computing, Inc. (OCI). All rights reserved.

PARTIAL CLIENT LIST

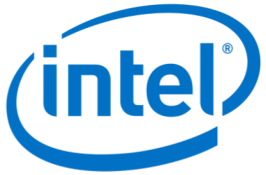


90%

REPEAT CLIENTS

© 2019, Object Computing, Inc. (OCI). All rights reserved.

OUR COMMUNITY OF PARTNERS



© 2019, Object Computing, Inc. (OCI). All rights reserved.

Grails

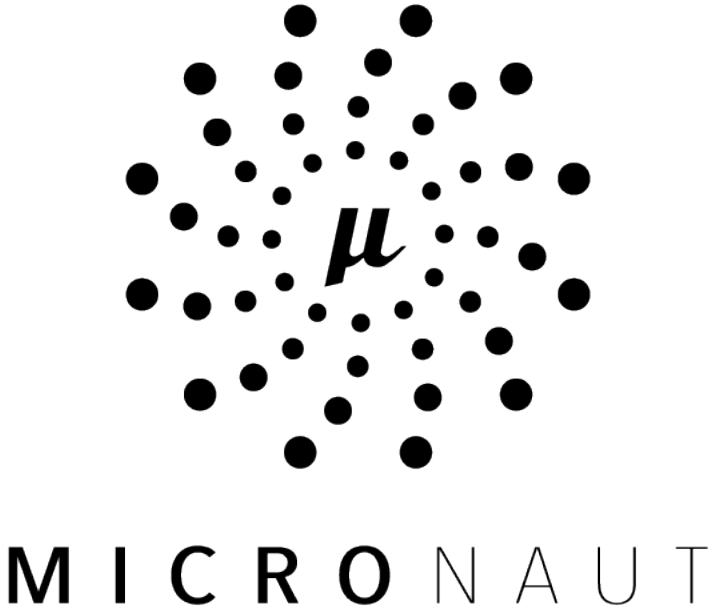
OCI | WE ARE SOFTWARE ENGINEERS.

OCI | HOME TO GRAILS

https://grails.org

© 2017, Object Computing, Inc. (OCI). All rights reserved.

Micronaut



https://micronaut.io

© 2017, Object Computing, Inc. (OCI). All rights reserved.

About Me

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Zachary Klein is a Senior Software Engineer at OCI. He has been practicing web development since 2010 and frontend development since 2015. He's a contributor to both the Grails and Micronaut frameworks, a conference speaker and an instructor in OCI's training practice.

Zachary's home base is in St Louis, MO, along with his wife, Beth, and their three children.

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Agenda

1. What is GraphQL?
    1. Overview
    2. Considerations & Limitations
2. Introducing GORM for GraphQL
    1. Overview
    2. Customization
    3. Testing

© 2018, Object Computing, Inc. (OCI). All rights reserved.

The Problem

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

REST is too **rigid** 😟

© 2019, Object Computing, Inc. (OCI). All rights reserved.

The Solution

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

GraphQL 😄

© 2019, Object Computing, Inc. (OCI). All rights reserved.

What is GraphQL?

OCI | WE ARE SOFTWARE ENGINEERS.

NOT a Graphing Database! 🤔

© 2018, Object Computing, Inc. (OCI). All rights reserved.

What is GraphQL?

OCI | WE ARE SOFTWARE ENGINEERS.

- Specification
- Schema Definition
- Introspective
- Application Layer Middleware
- Strongly-Typed

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Advantages of GraphQL

OCI | WE ARE SOFTWARE ENGINEERS.

Efficiency, **Efficiency**, *Efficiency* 🚀

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Advantages of GraphQL 👍

- Multiple operations simultaneously
- Only the data requested is returned
- Self documenting
- Client features to control response based on variables and data types
- One “endpoint” can support many use cases

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Disadvantages of GraphQL 👎

- Truly Dynamic Data Not Easy
- Recursive Data Structures
- Inflexible Security
- Time Intensive To Setup
- (Yet) Another Point Of Failure
- Client-side complexity

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Disadvantages of GraphQL 👎

- Truly Dynamic Data Not Easy
- Recursive Data Structures
- **Inflexible Security**
- **Time Intensive To Setup**
- (Yet) Another Point Of Failure
- Client-side complexity

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Demo GraphQL

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

GitHub API v4 is implemented with GraphQL:

https://developer.github.com/v4/

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Challenges implementing GraphQL

OCI | WE ARE SOFTWARE ENGINEERS.

- Generating a Schema
- Efficient queries
- Implementing HTTP layer (for web apps)

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Generating a Schema

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

1. Define your domain model

```
class Book {

    String title

    Date dateCreated
    Date lastUpdated

    static hasMany = [pages: Page]

    static constraints = {
    }
}
```

```
class Page {

    Integer number
    String text

    static belongsTo = [book: Book]

    static constraints = {
    }
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Generating a Schema

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

2. Define arguments for input

…for each argument type

```
newInputObject()
    .name( name: 'BookCreate')
    .field(newInputObjectField()
        .name( name: "title")
        .type(nonNull(Scalars.GraphQLString)))
    .field(newInputObjectField()
        .name( name: "pages")
        .type(list(newInputObject()
            .name( name: "PageCreateNested")
            .field(newInputObjectField()
                .name( name: "id")
                .type(Scalars.GraphQLLong))
            .field(newInputObjectField()
                .name( name: "number")
                .type(Scalars.GraphQLInt))
            .field(newInputObjectField()
                .name( name: "text")
                .type(Scalars.GraphQLString)).build())))
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Generating a Schema

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

3. Define each query/mutation

…handle versioning,

validation…

4. And so on…

```
newInputObject()
    .name( name: 'BookUpdate')
    .field(newInputObjectField()
        .name( name: "version")
        .type(Scalars.GraphQLLong))
    .field(newInputObjectField()
        .name( name: "title")
        .type(Scalars.GraphQLString))
    .field(newInputObjectField()
        .name( name: "pages")
        .type(list(newInputObject()
            .name( name: "PageUpdateNested")
            .field(newInputObjectField()
                .name( name: "id")
                .type(Scalars.GraphQLLong))
            .field(newInputObjectField()
                .name( name: "number")
                .type(Scalars.GraphQLInt))
            .field(newInputObjectField()
                .name( name: "text")
                .type(Scalars.GraphQLString)).build())))
```

New Field

Null Allowed

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Implementing HTTP layer

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Implement GraphQL best practices for:

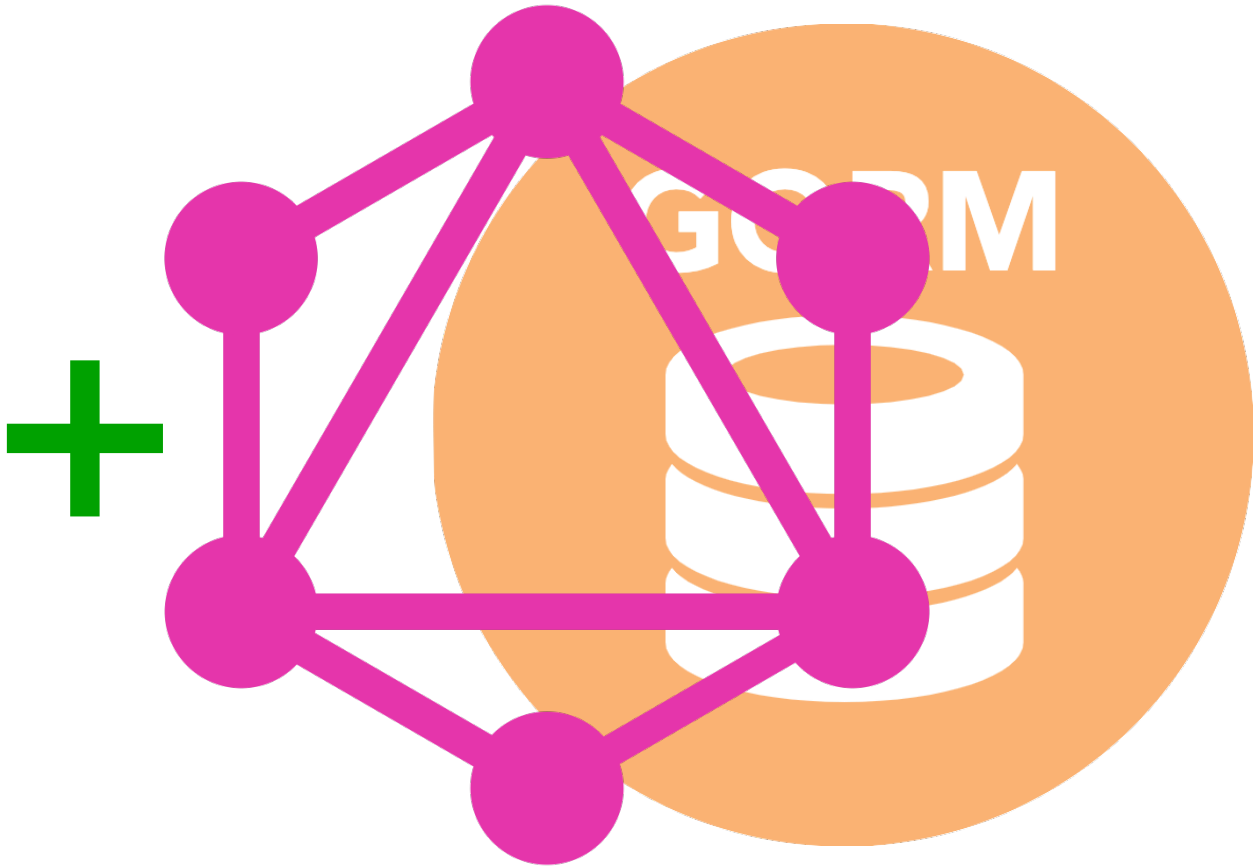
- Content Types
- Request Methods
- Parameter Names
- Endpoint URI
- Response Format

© 2019, Object Computing, Inc. (OCI). All rights reserved.

It’s a lot of work! 😴

GORM

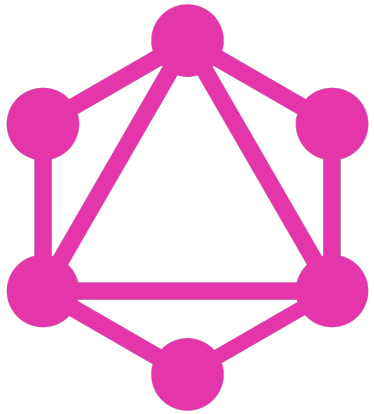
- **Groovy-based** data access &
  persistence framework
- Originally an ORM fram
  Hibernate
- Supports MongoDB, I
- Supports RxJava for
  applications
- Supports multiple data
- Usable in **Grails**, **Spring**
  Java applications generally


GORM

What is GORM for GraphQL?

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- GraphQL Schema-generation **library**
  - Also generates standard CRUD operations
- Grails (3 & 4) **plugin** with additional features
  - HTTP communication, GraphiQL browser, testing utilities, etc



© 2019, Object Computing, Inc. (OCI). All rights reserved.

GORM for GraphQL Features

OCI | WE ARE SOFTWARE ENGINEERS.

Core Library:

- Schema generation
- Query & Mutation generation
- Efficient Query building

Grails Plugin

- HTTP Communication
- Spring Integration
- GraphiQL Browser
- Data Binding
- Testing Support

© 2018, Object Computing, Inc. (OCI). All rights reserved.

Getting Started

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

1. Add `static graphql` property
2. GORM/GraphQL will generate 6 operations for each exposed domain class:
   - Create
   - Update
   - Delete
   - List
   - Show
   - Count

```
class Book {

    String title

    Date dateCreated
    Date lastUpdated

    static hasMany = [pages: Page]

    static constraints = {
    }

    static graphql = true
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Demo GORM for GraphQL

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Example Application:

https://github.com/grails-samples/graphql-demo

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizing the Schema

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- How do I customize the existing operations?
- How do I add my own operations?
- How do I add custom properties?
- How do I deprecate properties?
- How do I add a description for my properties?
- How do I supply my own queries?
- How do I change how validation errors are returned?
- How do I change the response for a delete?
- How do I …

© 2019, Object Computing, Inc. (OCI). All rights reserved.

GraphQL Customization Concepts

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- **Data Fetcher**: Responsible for retrieving all data
- **Interceptor**: Customize behavior of operations and fields within the schema
- **Data Binders**: Binds incoming arguments to persistence entities
- **Scalars**: Type conversion between custom types and the schema

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customization Opportunities

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

There are several areas where customizations can occur.

Persistent Entity Mapping

- Define Metadata (Description, etc)
- Add New Properties
- Create New Operations
- Override Data Fetchers For Properties
- Exclude Properties

Bean Post Processor (Plugin Only)

- Add Custom Scalar Types
- Register Data Fetchers For Domains
- Register Data Binders For Domains
- Register Interceptors

Bean Overrides (Plugin Only)

- Delete Response Handler
- Validation Errors Response Handler
- Many Others…

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizing the Schema

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Replace:

```
static graphql = true
```

With:

```
static graphql = GraphQLMapping.build {


}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizations - Return Types

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Return type of custom properties/operations/arguments/fields can be either:

1. `Class (Author)`: Instance of class
2. `List<Class> ([Author])`: Collection of instances
3. `PaginatedType`: Paginated collection
4. `String ("CustomType")`: Custom return type

```
add(name, type) {} // Adding a custom property
query(name, type) {} // Defining a new query operation
mutation(name, type) {} // Defining a new mutation operation
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizations - Custom Types

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- Custom types are a collection of fields.
- Custom types require a data fetcher to be provided in order to return the desired data to GraphQL.
- The defined fields follow the same rules for return types and can be a custom type themselves.

```
class Author {

    LocalDate birthDay

    Map getAge() {
        LocalDate now = LocalDate.now()

        [days: ChronoUnit.DAYS.between(birthDay, now),
         months: ChronoUnit.MONTHS.between(birthDay, now),
         years: ChronoUnit.YEARS.between(birthDay, now)]
    }

    static graphql = GraphQLMapping.build {
        add( 'age', typeName: 'Age') {
            input input: false
            type {
                field('days', Long)
                field('months', Long)
                field('years', Long)
            }
        }
    }
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizations - Metadata

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Metadata can be defined either via the mapping DSL or the @GraphQL annotation

Examples

Adding a description to a property (Mapping DSL):

```
property( name: "name", description: "The name of the Author")
```

And with the annotation:

```
@GraphQL("The name of the author")
String name
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizations - Custom Operations

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- Operations of of two types: **query** and **mutation**
- Both type require a **data fetcher**

```
query( name: 'usersByRole', [User]) {
    argument('role', Long)
    dataFetcher(new UsersByRoleDataFetcher())
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizations - Custom Operations

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

```
@CompileStatic
class UsersByRoleDataFetcher extends EntityDataFetcher<List<User>> {

    UsersByRoleDataFetcher() {
        super(User.gormPersistentEntity)
    }

    @GrailsCompileStatic
    @Override
    protected List executeQuery(DataFetchingEnvironment environment, Map queryArgs) {
        Role role = Role.load((Serializable) environment.getArgument( name: 'role'))
        def users = UserRole.where { role == role }.property( property: 'user')
        User.where {
            id in users
        }.list(queryArgs)
    }
}
```

Provided By Library

Type being queried

Join Properties

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Data Fetchers - Context

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- Data Fetchers are not part of the Spring application context
- Not request-aware
- Data Fetchers have a **context** which can be manipulated via an exposed bean
- Default context is a **Map** with a single key, “locale” (locale of the current HTTP request)

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Data Fetchers - Context

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

```
class MyGraphQLContextBuilder extends DefaultGraphQLContextBuilder {

    @Autowired
    SpringSecurityService springSecurityService

    @Override
    Map buildContext(GrailsWebRequest request) {
        Map context = super.buildContext(request)
        context.springSecurityService = springSecurityService
        context
    }
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customization - Interceptors

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- GraphQL uses a single endpoint
- Security rules cannot by applied based on URL
- Interceptors provide an opportunity to perform authorization checking
- Best practice: return null for data/operations that are not allowed
- 2 basic types of interceptors:
  - Interceptor executed prior to data fetcher execution
  - Interceptor executed prior to schema generation (low-level)

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customization - Interceptors

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

```
class AuthorUpdateInterceptor extends BaseGraphQLFetcherInterceptor {

    @Override
    boolean onMutation(DataFetchingEnvironment environment, GraphQLDataFetcherType type) {
        if (type == GraphQLDataFetcherType.UPDATE) {
            Author author = Author.get((Long) environment.getArgument( name: "id"))
            Map context = (Map) environment.context
            SpringSecurityService springSecurityService = (SpringSecurityService) context.springSecurityService
            springSecurityService.currentUser == author.user
        } else {
            true
        }
    }
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Interceptor Registration

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

1. Define a bean post processor class and access the `GraphQLInterceptorManager`.
2. Register your custom interceptor and any domain classes that should trigger it

```
class MyGraphQLCustomizer extends GraphQLPostProcessor {

    @Override
    void doWith(GraphQLInterceptorManager interceptorManager) {
        interceptorManager.registerInterceptor(Author, new AuthorUpdateInterceptor())
    }
}
```

3. Register this class as a bean in resources.groovy.

```
beans = {
    myGraphQLCustomizer(MyGraphQLCustomizer)
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Customizations - Data Binding

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

```
@CompileStatic
class UserDataBinder extends GrailsGraphQLDataBinder {

    @Override
    void bind(Object object, Map data) {
        Integer first = (Integer)data.remove( key: 'firstNumber')
        Integer second = (Integer)data.remove( key: 'secondNumber')
        if (first != null && second != null) {
            data.put('addedNumbers', first + second)
        }
        super.bind(object, data)
    }
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Data Binder Registration

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

1. Define a bean post processor class and access the `GraphQLInterceptorManager`.
2. Register your custom interceptor and any domain classes that should trigger it

```
class MyGraphQLCustomizer extends GraphQLPostProcessor {

    @Override
    void doWith(GraphQLDataBinderManager dataBinderManager) {
        dataBinderManager.registerDataBinder(User, new UserDataBinder())
    }
}
```

3. Register the class as a bean in resources.groovy.

```
beans = {
    myGraphQLCustomizer(MyGraphQLCustomizer)
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Testing Utilities

OCI | WE ARE SOFTWARE ENGINEERS.

The Grails Plugin provides two testing helpers:

1. Functional test trait
2. Unit-testing support for schema generation

© 2017, Object Computing, Inc. (OCI). All rights reserved.

Functional Test Trait

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

```
@Integration
class PetIntegrationSpec extends Specification implements GraphQLSpec {

    void "test calling graphql"() {
        when:
        def resp = graphQL.graphql( requestBody: """
            mutation {
                petCreate(pet: {
                    name: "Butch",
                    type: DOG
                }) {
                    id
                }
            }
        """)

        def result = resp.json

        then:
        result.data.petCreate.id == 1
    }
}
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Schema Testing

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

```
class SchemaSpec extends Specification implements GraphQLSchemaSpec {

    @Shared @AutoCleanup HibernateDatastore hibernateDatastore
    @Shared GraphQLSchema schema
    @Shared GraphQLObjectType queryType
    @Shared GraphQLObjectType mutationType

    void setupSpec() {
        hibernateDatastore = new HibernateDatastore(
                DatastoreUtils.createPropertyResolver(
                        Collections.singletonMap(Settings.SETTING_DB_CREATE, 'create-drop')),
                GeneralPackage.getPackage(), HibernatePackage.getPackage())
        schema = new Schema(hibernateDatastore.mappingContext).generate()
        queryType = schema.queryType
        mutationType = schema.mutationType
    }
```

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Resources

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

- Documentation: https://grails.github.io/gorm-graphql/snapshot/guide/index.html
- Grails Guide: https://guides.grails.org/gorm-graphql-with-react-and-apollo/guide/index.html
- GraphQL website: https://www.graphql.org
- Grails Sample code: https://github.com/grails-samples/graphql-demo
- Micronaut sample code: https://github.com/jameskleeh/micronaut-gorm-graphql-example

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Grails Application Forge: http://start.grails.org

GRAILS APPLICATION FORGE

Project Type:
Application

Name your application:
myapp

Version:
3.3.5

Profile:
web

Pick your Features:
- [x] asset-pipeline
- [x] events
- [x] geb
- [ ] geb2
- [x] gsp
- [ ] hibernate4
- [x] hibernate5
- [ ] json-views
- [ ] less-asset-pipeline
- [ ] markup-views
- [ ] mongodb
- [ ] neo4j
- [ ] rx-mongodb

Generate

© 2019, Object Computing, Inc. (OCI). All rights reserved.

Grails Guides: http://guides.grails.org

GUIDES

LATEST GUIDES

TWITTER OAUTH WITH GRAILS 3 AND SPRING SECURITY REST
Apr 30, 2018 - Advanced Grails
Read More

GORM LOGICAL DELETE
Apr 09, 2018 - GORM
Read More

JAVAMELODY MONITORING WITH GRAILS 3
Apr 02, 2018 - Grails + Devops
Read More

BUILDING A VUE.JS APP WITH GRAILS
Mar 26, 2018 - Grails + Vue.js
Read More

SEARCH

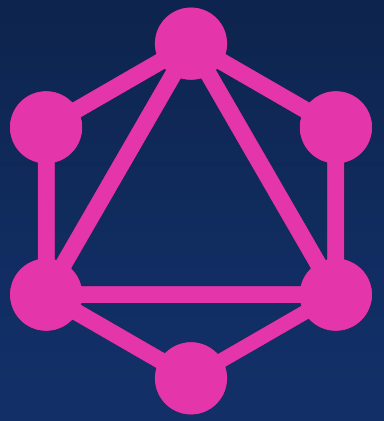
GRAILS TRAINING

Course	Date(s)	Instructor(s)	Hour(s)
Micronaut Deep Dive at GR8Conf US	Jul 23 - Jul 24	Brown	12
Introduction to Grails Security	May 17 - May 18	del Amo Caballero	6
GROOVY 2.5 UPDATE	May 18	King	1

© 2019, Object Computing, Inc. (OCI). All rights reserved.

OBJECT COMPUTING

Thank you!



© 2019, Object Computing, Inc. (OCI). All rights reserved.

Q&A

kleinz@objectcomputing.com | Twitter: @ZacharyAKlein | Github: ZacharyKlein

LEARN MORE ABOUT OCI EVENTS AND TRAINING

OBJECT COMPUTING
HOME TO GRAILS & MICRONAUT

Events:

- objectcomputing.com/events

Training:

- objectcomputing.com/training
- grailstraining.com
- micronauttraining.com

Or email info@ocitraining.com to schedule a custom training program for your team online, on site, or in our state-of-the-art, Midwest training lab.

© 2019, Object Computing, Inc. (OCI). All rights reserved.

OBJECT COMPUTING

HOME TO GRAILS & MICRONAUT

CONNECT WITH US

1+ (314) 579-0066

@objectcomputing

objectcomputing.com

© 2019, Object Computing, Inc. (OCI). All rights reserved.