Основи GraphQL API: розробка засобами Java

Вступ до GraphQL. Порівняння з REST. Специфікація GraphQL

Стельмащук Віталій Володимирович

Львівський національний університет імені Івана Франка
Кафедра інформаційних систем

10 лютого 2025 р.

Короткий зміст

Організація курсу

- Лекційний курс
- Захист практичних завдань - 80 балів
  - 4 завдання по 10 балів
  - завдання-проект - 40 балів
- Теоретичний тест за матеріалами лекцій - 20 балів
- Залік

Короткий зміст курсу

- Порівняння з REST. Специфікація GraphQL
- Бібліотека graphql-java як реалізація специфікації GraphQL
- Фреймворк Spring for GraphQL для побудови GraphQL API
- Netflix DGS Framework для побудови GraphQL API
- GraphQL API та проблема N+1 запиту. Data Loaders в graphql-java, Spring for GraphQL та Netflix DGS Framework
- Найкращі практики в побудові GraphQL API
- Обробка помилок в GraphQL API
- Користувацькі скаляри в GraphQL API

Короткий зміст курсу

- Підписки GraphQL в Spring for GraphQL та Netflix DGS Framework
- Безпека GraphQL API. Поєднання Spring Security та KeyCloak для реалізації автентифікації та авторизації (RBAC) у GraphQL API
- Тестування GraphQL API засобами Spring for GraphQL та Netflix DGS Framework
- Виклик GraphQL API клієнтами Java: засоби Spring for GraphQL та Netflix DGS Framework
- Логування та моніторинг викликів GraphQL API: засоби Spring for GraphQL та Netflix DGS Framework

Основна література

1. Christudas B. A. Java Microservices and Containers in the Cloud: With Spring Boot, Kafka, PostgreSQL, Kubernetes, Helm, Terraform and AWS EKS / B. A. Christudas. – Apress Berkeley, CA, 2024. – 843p.
2. Sharma S. Modern API development with Spring 6 and Spring Boot 3: design scalable, viable, and reactive APIs with REST, gRPC, and GraphQL using Java 17 and Spring Boot 3, 2nd ed. / S. Sharma. – Packt Publishing, 2023. – 494p.
3. GraphQL Java Documentation – Електронний ресурс. Режим доступу: https://www.graphql-java.com/documentation/getting-started
4. Spring for GraphQL Reference Documentation – Електронний ресурс. Режим доступу: https://www.graphql-java.com/documentation/getting-started
5. Netflix DGS Framework Documentation – Електронний ресурс. Режим доступу: https://netflix.github.io/dgs/

Додаткова література

1. Buna S. GraphQL in Action / S. Buna. – Manning, 2021. – 384p.
2. Deinum M. Spring Boot 3 Recipes: A Problem-Solution Approach for Java Microservices and Cloud-Native Applications, 2nd ed. / M. Deinum. – Apress Berkeley, CA, 2024. – 524p.
3. GraphQL Specification – Електронний ресурс. Режим доступу: https://spec.graphql.org/October2021/

Що таке GraphQL?

- GraphQL – це декларативна мова запитів для серверних API та середовище виконання для цих запитів.
- Розроблено Facebook в 2012, open-source в 2015.
- Дозволяє клієнтам надсилати запити на отримання тільки тих даних, які їм необхідні.
- Популярний підхід для розробки API, що використовуються аплікаціями для мобільних телефонів.

Коротка історія GraphQL

- 2011: Facebook зіткнувся з викликами щодо оптимізації продуктивності мобільного веб-сайту.
- API мали труднощі з ієрархічними та рекурсивними даними, що спричиняло неефективність.
- Швидкість мобільного інтернету була низькою (у деяких регіонах вимірювалася в Кб/с).
- 2012: Інженери Лі Байрон, Ден Шефер і Нік Шрок створили GraphQL.
- Спочатку розроблявся для функції новинної стрічки Facebook.
- Використання розширилося на всю інфраструктуру Facebook.
- 2015: Опубліковано GraphQL специфікацію, open-source реалізація на JavaScript.
- Згодом з'явилися реалізації для інших мов, зокрема Java.

Документальний фільм про створення GraphQL

Порівняння GraphQL API з REST API

- E-commerce UI app via REST API backend
- Користувач логіниться в аплікацію і автоматично переходить на сторінку зі списком продуктів, які він може придбати
- REST API запит на отримання інформації про поточного користувача:

  Request:

```
GET /api/users/123
```

  Response:

```
{
        "id": 123,
        "name": "John Doe",
        "email": "john@example.com"
}
```

REST API: Отримання списку продуктів та стану кошика користувача

Request:

```
GET /api/products
```

Response:

```
[
        { "id": 1, "name": "Laptop", "price": 1000 },
        { "id": 2, "name": "Mouse", "price": 50 }
]
```

Request:

```
GET /api/users/123/cart
```

Response:

```
[
        { "productId": 1, "quantity": 1 },
        { "productId": 2, "quantity": 2 }
]
```

GraphQL API: Єдиний запит для отримання даних

```
query {
        user(id: 123) {
                id
                name
                email
                cart {
                        product {
                                id
                                name
                                price
                        }
                        quantity
                }
        }
        products {
                id
                name
                price
        }
}
```

GraphQL API: Приклад відповіді бекенду

```
{
        "data": {
                "user": {
                        "id": 123,
                        "name": "John Doe",
                        "email": "john@example.com",
                        "cart": [
                        {
                                "product": {
                                        "id": 1,
                                        "name": "Laptop",
                                        "price": 1000
                                },
                                "quantity": 1
                        }
                        ]
                }
        }
```

GraphQL API: Приклад відповіді бекенду (продовження)

```
                "products": [
                        {
                                "id": 1,
                                "name": "Laptop",
                                "price": 1000
                        },
                        {
                                "id": 2,
                                "name": "Mouse",
                                "price": 50
                        }
                ]
        }
}
```

Специфікація GraphQL. Основні принципи

Специфікація GraphQL

- **Орієнтованість на продукт:** Зосереджений на потребах фронтенд-розробників і їхніх уявленнях.
- **Ієрархічність:** Запити та відповіді мають природну ієрархічну структуру.
- **Сильна типізація:** Кожен GraphQL-сервіс визначає сувору систему типів для забезпечення коректності запитів.
- **Відповідь, визначена клієнтом:** Клієнти контролюють структуру відповідей, вказуючи, які саме дані їм потрібні.
- **Інтроспективність:** GraphQL дозволяє робити запити про власну систему типів для полегшення розробки.

Мова запитів GraphQL. Види операцій

- **Query (Запит):** Використовується для отримання даних.
- **Mutation (Мутація):** Використовується для зміни даних (створення, оновлення, видалення).

```
mutation {
        updateUser(id: 123, name: "John Doe") {
                id
                name
        }
}
```

- **Subscription (Підписка):** Використовується для отримання оновлень у реальному часі.

```
subscription {
        newMessage {
                content
                sender
        }
}
```

SelectionSets у мові запитів GraphQL

SelectionSet - це набір полів, які клієнт запитує в GraphQL.

```
query {
        user(id: 123) {
                name
                email
                cart {
                        product {
                                name
                                price
                        }
                }
        }
}
```

Джерело: GraphQL Специфікація

Поля у мові запитів GraphQL

Field (Поле) - це базова одиниця запиту в GraphQL.

```
query {
        user(id: 123) {
                name
                email
        }
}
```

Джерело: GraphQL Специфікація

Аргументи у мові запитів GraphQL

Arguments (Аргументи) - використовуються для передачі параметрів у запитах.

```
query {
        user(id: 123) {
                name
                email
        }
}
```

Джерело: GraphQL Специфікація

Псевдоніми полів у GraphQL

Field Aliases (Псевдоніми полів) - для зміни назв полів у JSON.

```
query {
        user1: user(id: 123) {
                name
        }
        user2: user(id: 456) {
                name
        }
}
```

```
{
        "data": {
                "user1": {
                        "name": "John Doe",
                        "email": "john@example.com"
                },
                "user2": {
                        "name": "Jane Smith",
                        "email": "jane@example.com"
                }
        }
```

Система типів GraphQL. GraphQL Schema

- Root Operation Types: **Query**, **Mutation**, **Subscription**.
- Types:
  - ScalarType
  - ObjectType
  - InterfaceType
  - UnionType
  - EnumType
  - InputObjectType
- Scalars and Enums form the leaves in response trees.
- Object types form intermediate levels in the response.
- 2 abstract types: Interfaces and Unions. Interface defines a list of fields. Union defines a list of possible types.
- InputObjectType is used for complex data structs as inputs to GraphQL field arguments or variables.

GraphQL Schema: User and Product

Schema Definition:

```
type User {
        id: ID!
        name: String!
        email: String!
        cart: [CartItem!]
}

type Product {
        id: ID!
        name: String!
        price: Float!
}
```

GraphQL Schema: CartItem and Query

Schema Definition:

```
type CartItem {
        product: Product!
        quantity: Int!
}

type Query {
        user(id: ID!): User
        products: [Product!]
}
```

InputType для пошуку товарів

Оновлення схеми: Додається новий InputType `ProductFilter`, який дозволяє фільтрувати товари за назвою або ціновим діапазоном.

```
input ProductFilter {
        name: String
        minPrice: Float
        maxPrice: Float
}

type Query {
        user(id: ID!): User
        products(filter: ProductFilter): [Product!]!
}
```

Приклад реального GraphQL API

GitHub GraphQL API:
https://docs.github.com/en/graphql/overview/explorer

Дякую за вашу увагу!

Запитання?