Spring for GraphQL

SpringOne Tour
2022

vmware

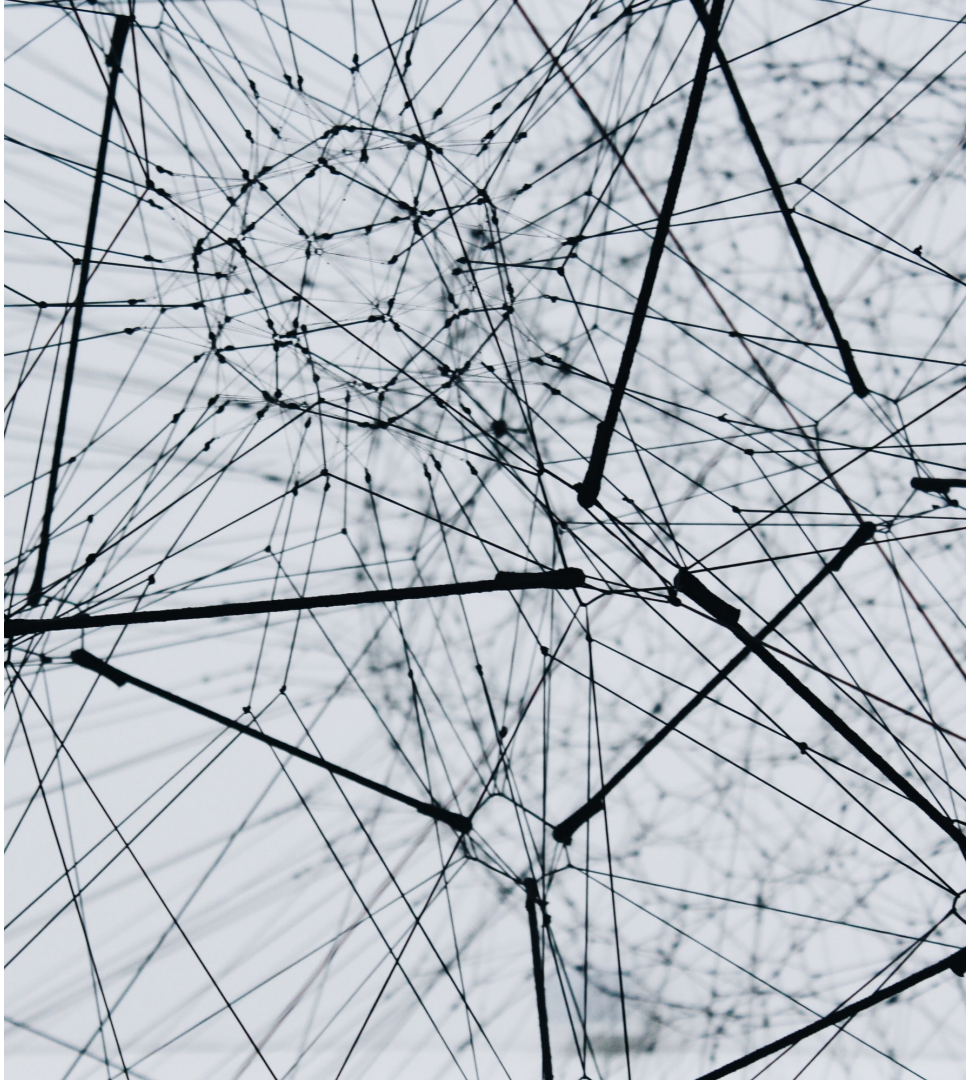
About Me

- Husband & Father
- Cleveland
- Software Development 20+ Years
- Spring Developer Advocate
- Content Creator (www.danvega.dev)
  - Blogger
  - YouTuber
  - Course Creator

spring

Agenda

- Why GraphQL?
- What is GraphQL?
- GraphQL Java
- Spring for GraphQL
- Q&A



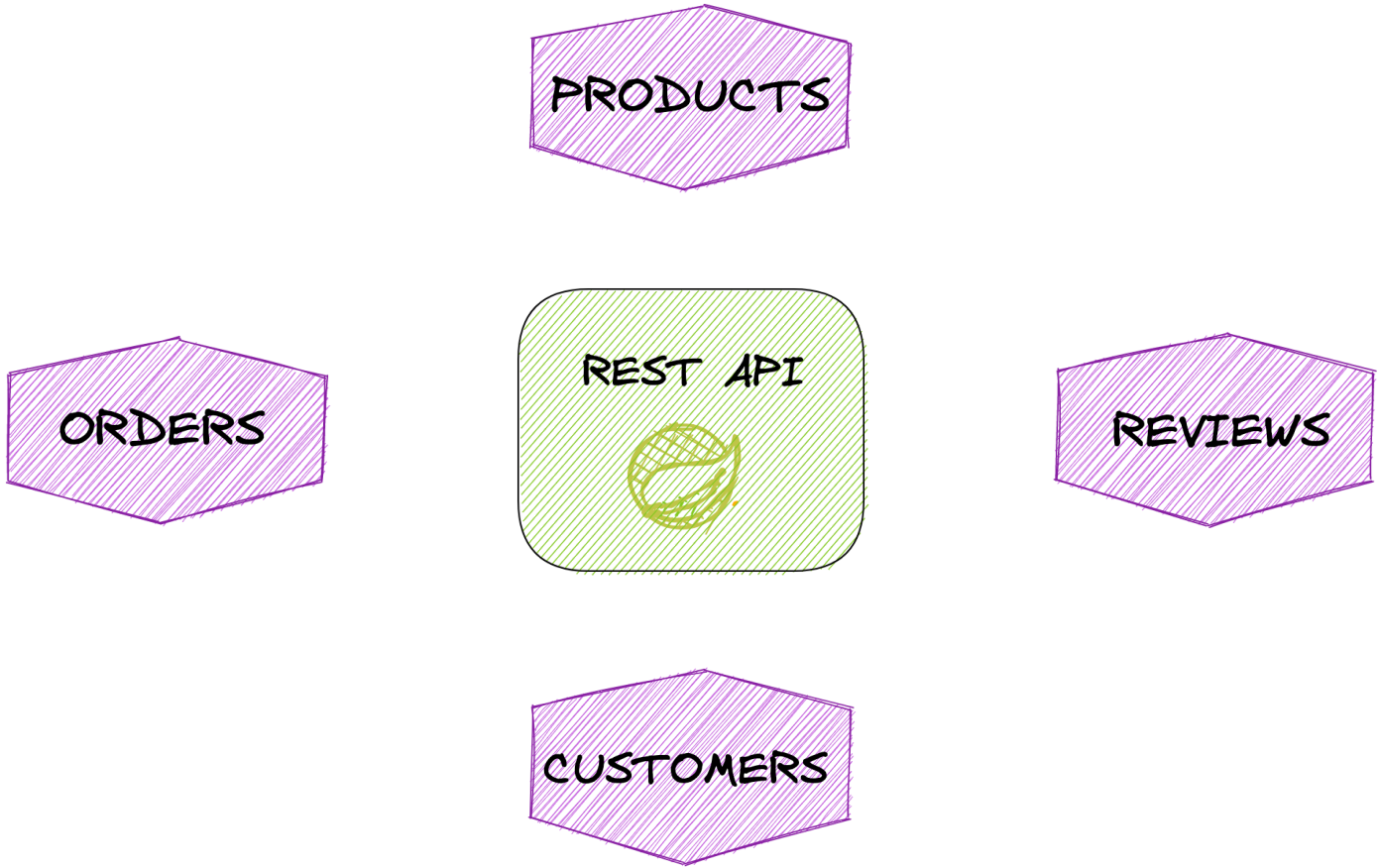
spring

WaffleCorp

- WaffleCorp Products
    - Direct to Consumer
    - Strategic Partnerships
- WaffleCorp e-commerce service
    - Monolithic Spring Boot MVC
    - REST API
    - Vue Front End
- Open up our REST APIs
    - Partner Microservices
    - iOS and Android applications
    - IoT Applications (Smart Toaster)

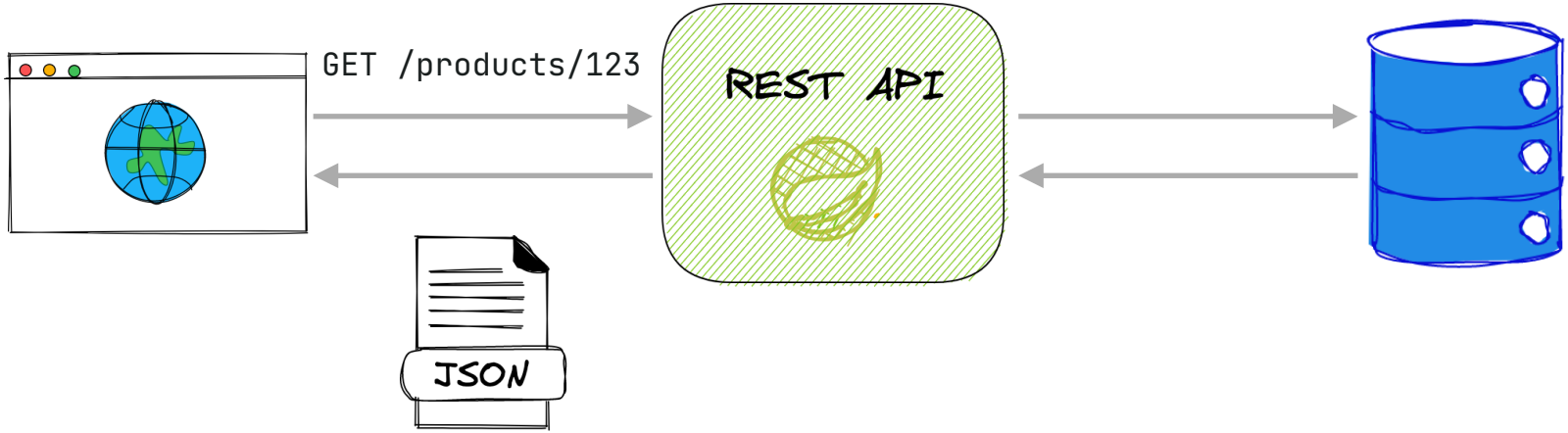
Why GraphQL?

Current Architecture



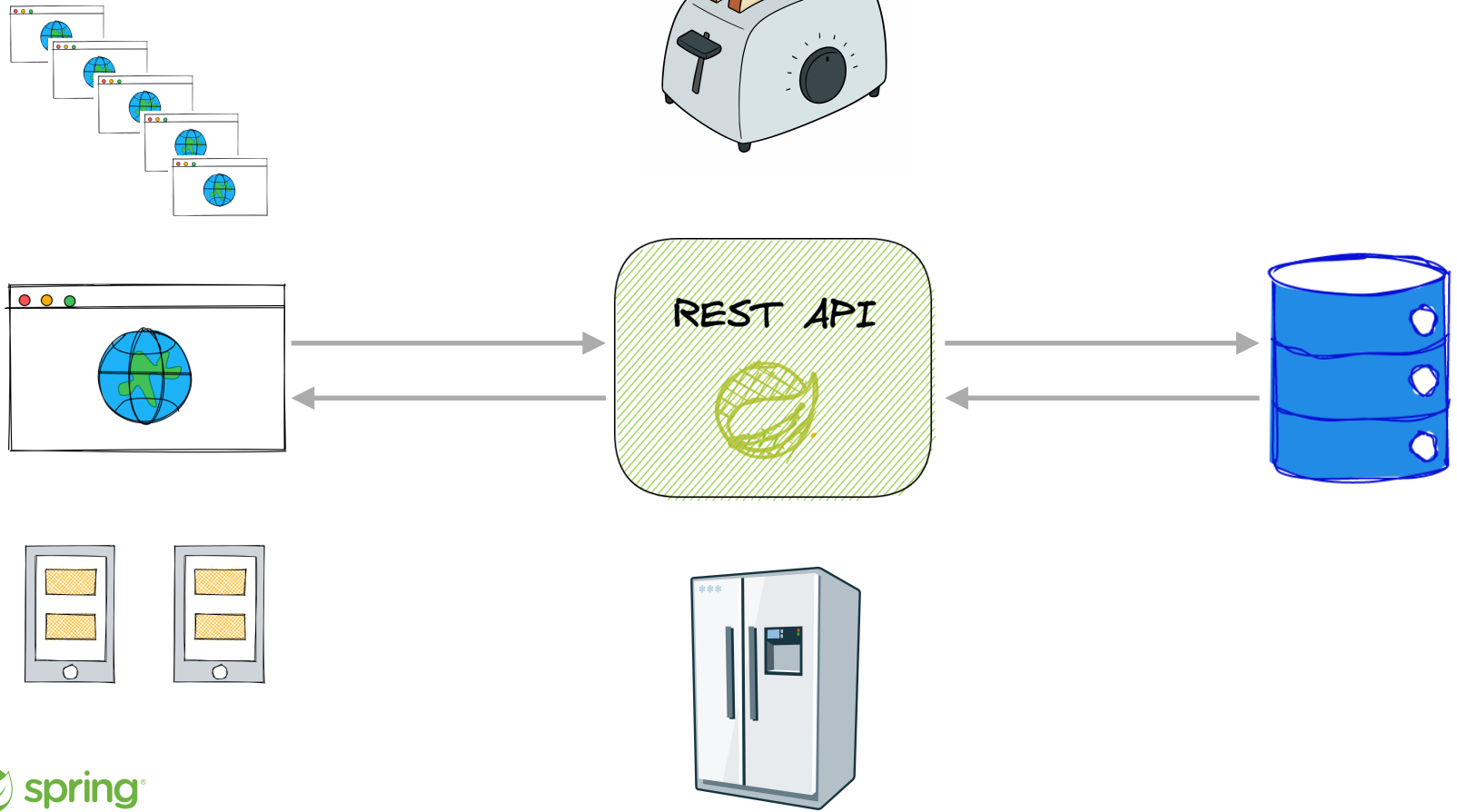
spring

Current Architecture



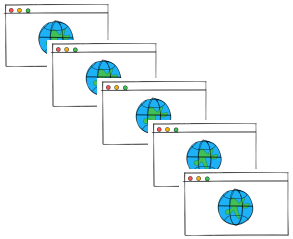
spring

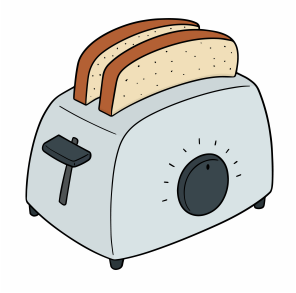
Current Architecture

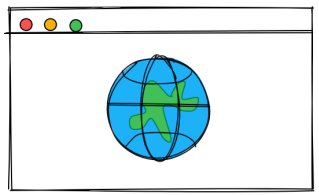


spring

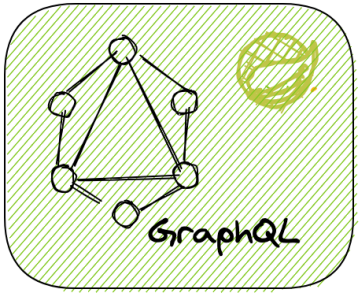
GraphQL Architecture

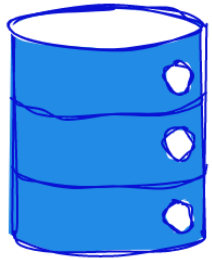


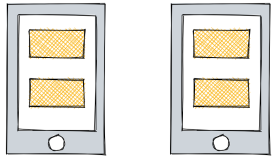


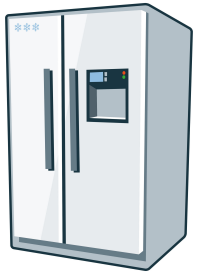


POST /graphql



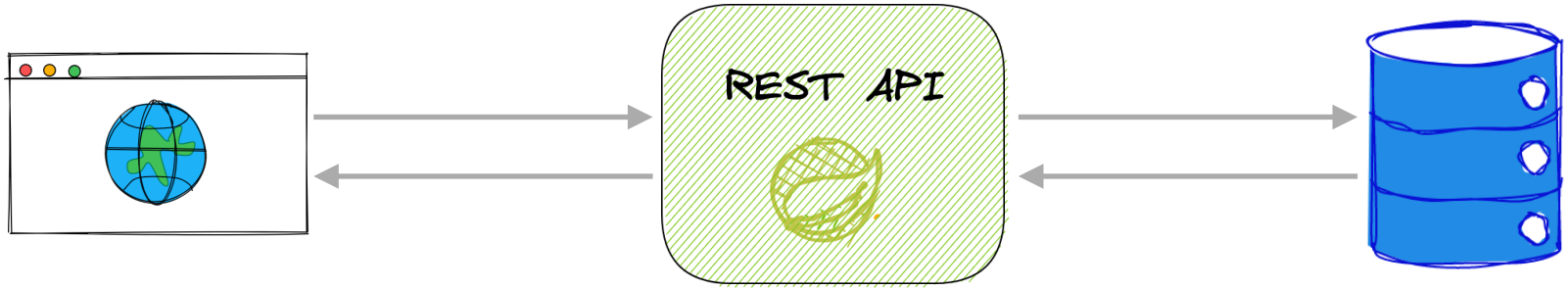






spring

Current Architecture



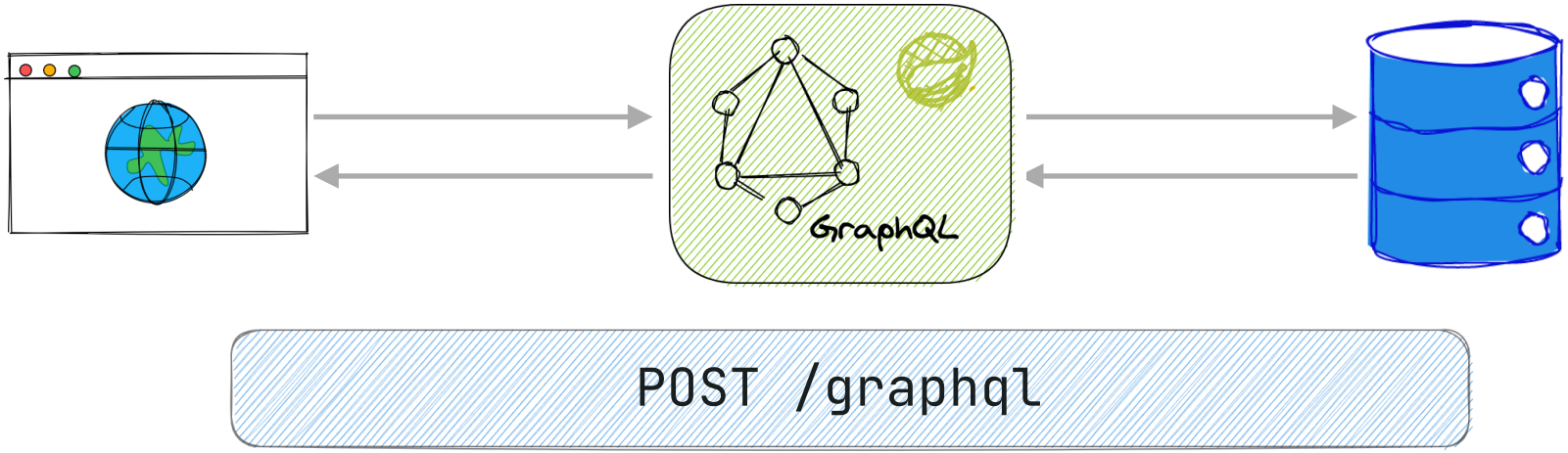
GET /products/123

GET /products/123/related

GET /products/123/reviews

spring

Current Architecture



What is GraphQL?

GraphQL is a query language for your API, and a server-side runtime for executing queries using a type system you define for your data. GraphQL **isn't tied to any specific database or storage engine** and is instead backed by your existing code and data.

GraphQL is a technology for client server data exchange

- Alternative to REST APIs
- Provides a really great developer experience
- Created by Facebook in 2012
  - Open Sourced in 2015
  - Governed by a non profit foundation today
- Client + Server: 🤝
  - Client Side Query Language
  - Server Side execution engine

Client

- Custom GraphQL Query Language
- Client crafted query based on their needs
- Responses return only what client requests
- Developer Tooling

Server

- GraphQL Schema defines your API
- Schema based on **simple** static type system
- Schema Definition Language (SDL) is used to describe a schema.
- Implemented in specific languages based on the GraphQL specification.

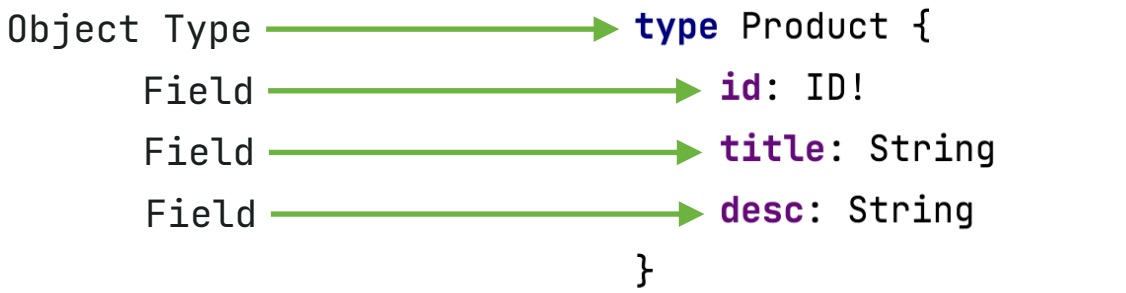
Type System

```
type Product {
    id: ID!
    title: String
    desc: String
}

type Order {
    id: ID!
    product: Product
    qty: Int
    customer: Customer
    orderedOn: Date
    status: OrderStatus
}
```

Object Type

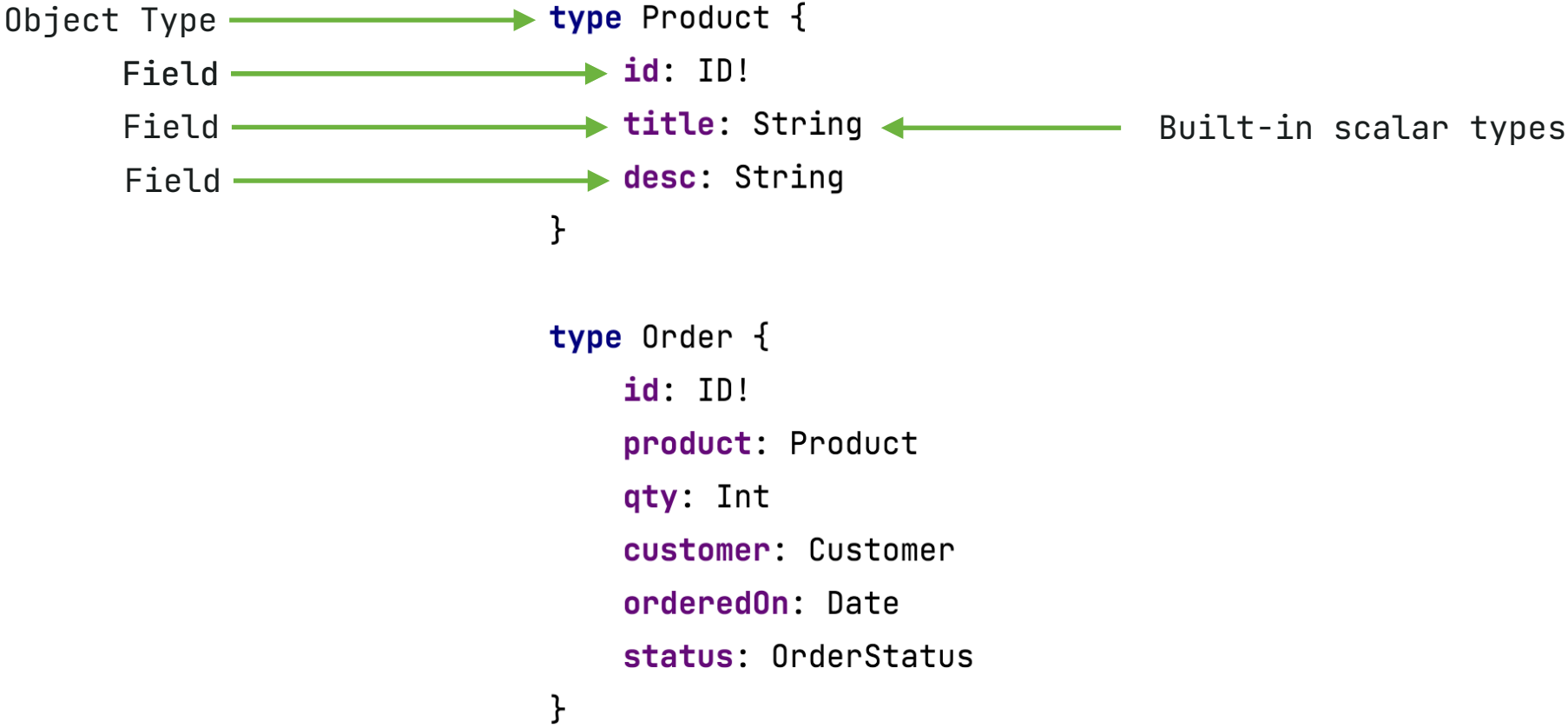
```
type Product {
    id: ID!
    title: String
    desc: String
}

type Order {
    id: ID!
    product: Product
    qty: Int
    customer: Customer
    orderedOn: Date
    status: OrderStatus
}
```

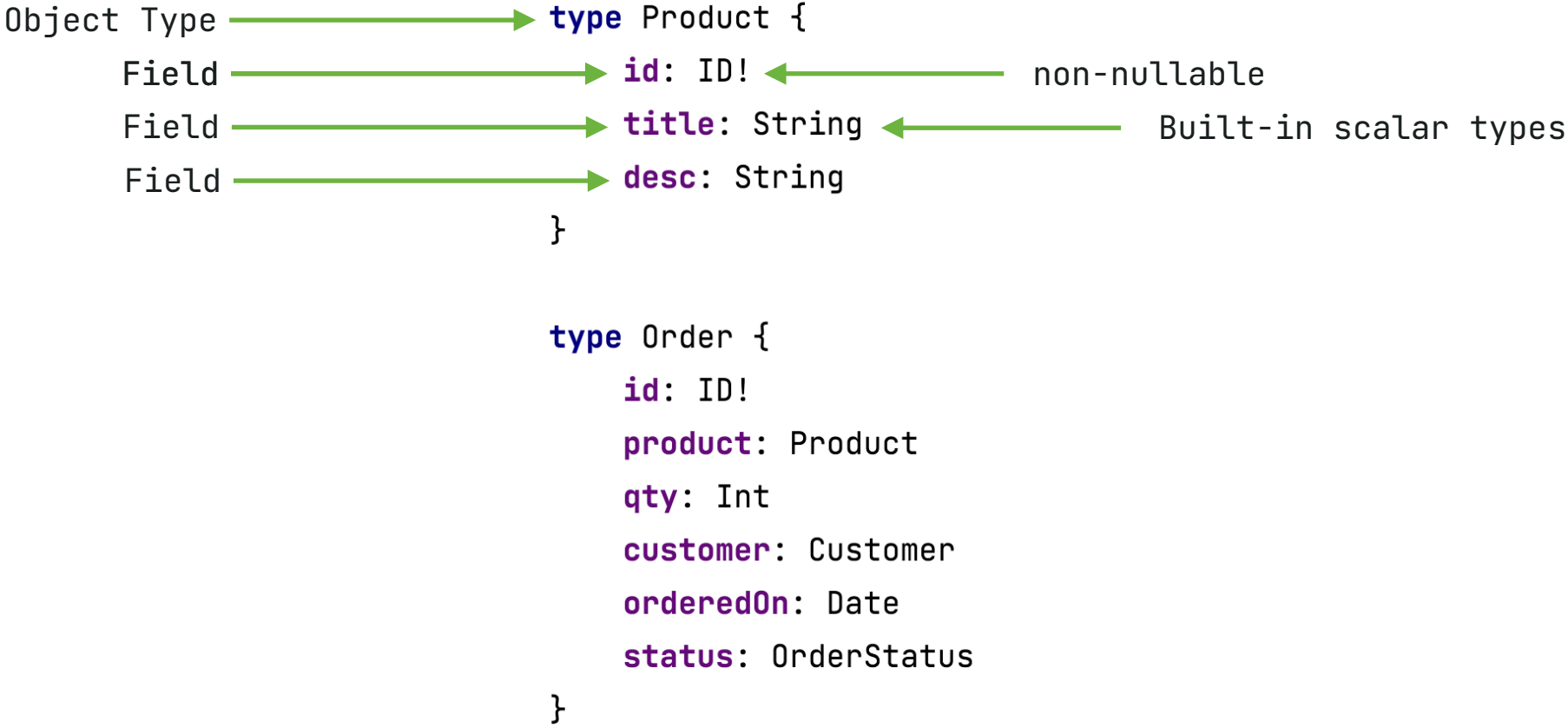
spring



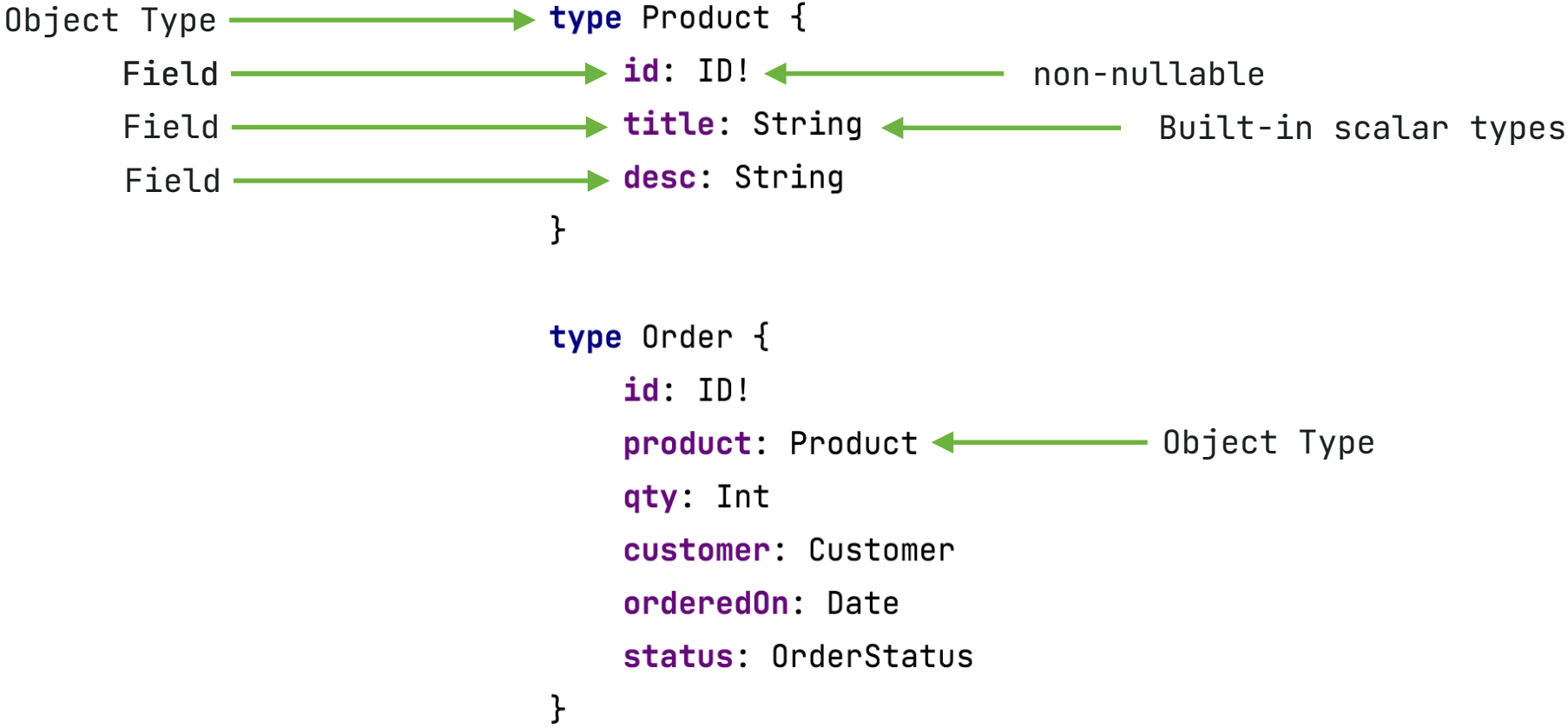

```
type Product {
    id: ID!
    title: String
    desc: String
}

type Order {
    id: ID!
    product: Product
    qty: Int
    customer: Customer
    orderedOn: Date
    status: OrderStatus
}
```

spring

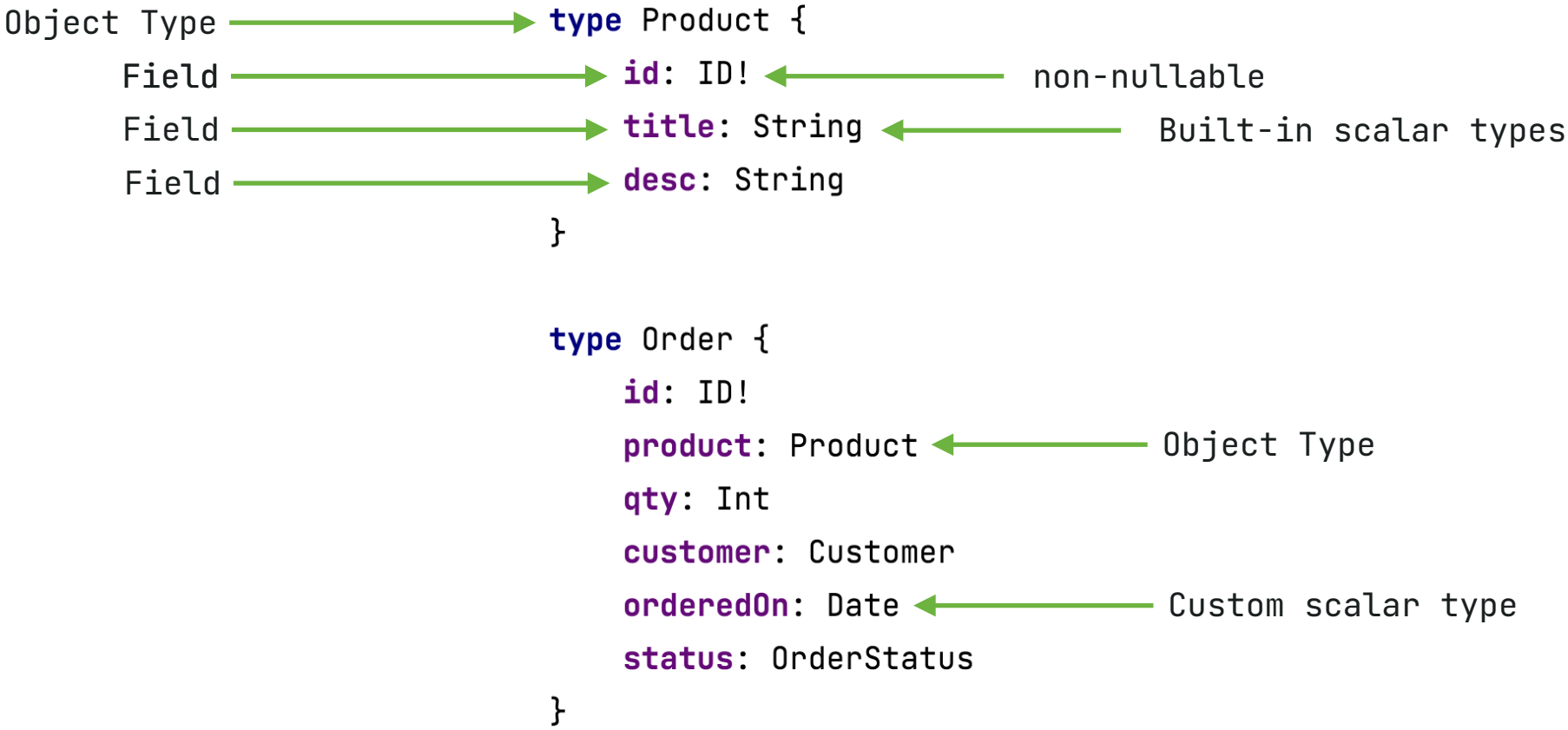

Object Type
Field
Field
Field
type Product {
id: ID!
title: String
desc: String
}
Built-in scalar types
type Order {
id: ID!
product: Product
qty: Int
customer: Customer
orderedOn: Date
status: OrderStatus
}


Object Type
Field
Field
Field
type Product {
id: ID!
title: String
desc: String
}
non-nullable
Built-in scalar types
type Order {
id: ID!
product: Product
qty: Int
customer: Customer
orderedOn: Date
status: OrderStatus
}

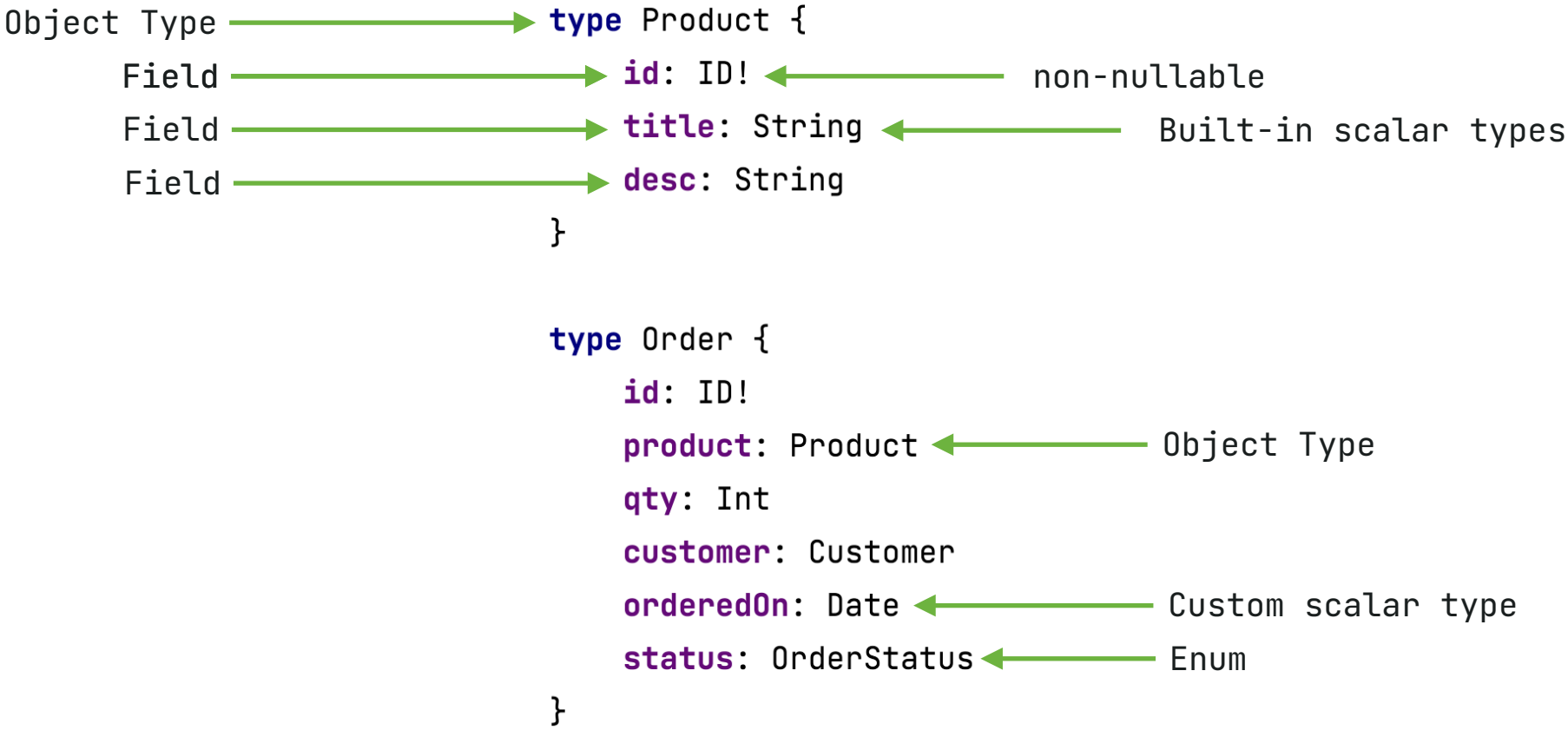
spring


Object Type
Field
Field
Field
type Product {
id: ID!
title: String
desc: String
}
non-nullable
Built-in scalar types
type Order {
id: ID!
product: Product
qty: Int
customer: Customer
orderedOn: Date
status: OrderStatus
}
Object Type

spring


Object Type
Field
Field
Field
type Product {
id: ID!
non-nullable
title: String
Built-in scalar types
desc: String
}
type Order {
id: ID!
product: Product
Object Type
qty: Int
customer: Customer
orderedOn: Date
Custom scalar type
status: OrderStatus
}

spring


Object Type
Field
Field
Field
type Product {
id: ID!
non-nullable
title: String
Built-in scalar types
desc: String
}
type Order {
id: ID!
product: Product
Object Type
qty: Int
customer: Customer
orderedOn: Date
Custom scalar type
status: OrderStatus
Enum
}

spring

GraphQL comes with a set of default scalar types out of the box:

- `Int`: A signed 32-bit Integer.
- `Float`: A signed double-precision floating point value.
- `String`: A UTF-8 character sequence.
- `Boolean`: true or false.
- `ID`: The ID scalar type represents a unique identifier, often used to refetch an object or as the key for a cache. The ID type is serialized in the same way as a String; however, defining it as an `ID` signifies that it is not intended to be human-readable.

Operation Types

Most types in your schema will just be normal object types, but there are three types that are special within a schema:

```
type Product {
    id: ID!
    title: String
    desc: String
}

type Query {
    allProducts: [Product]!
    getProduct(id: ID): Product!
}
```

```
type Product {
    id: ID!
    title: String
    desc: String
}

type Query {
    allProducts: [Product]!
    getProduct(id: ID): Product!
}
```

Query Name

spring

```
type Product {
    id: ID!
    title: String
    desc: String
}

type Query {
    allProducts: [Product]!
    getProduct(id: ID): Product!
}
```

spring

GraphiQL Demo

GraphQL Java

GraphQL Java is the GraphQL implementation for Java

- Started mid 2015
  - Andreas Marek, Software Architect at Atlassian
  - Bootiful Podcast Interview with Andreas
- Pure Execution Engine: No HTTP or IO. No high level abstractions
- Active on Github
  - 5.5k Stars / 1000 Forks
- Adds Custom Scalar Types
- https://www.graphql-java.com/

← Tweet

GraphQL Java
@graphql_java

Twitter is now powered by #GraphQL #Java. Serving 500k requests per second. github.com/graphql-java/g…
Thanks a lot for the public feedback @Twitter @jbell

github.com
twitter + graphql-java · Discussion #2591 · graphql-java/graphql-java
Hi team! Twitter is wrapping up its migration to graphql-java and I wanted to share some info about how we use graphql and why we chose to migrate our …

4:24 PM · Oct 18, 2021 · Twitter for iPhone

spring

Spring for GraphQL

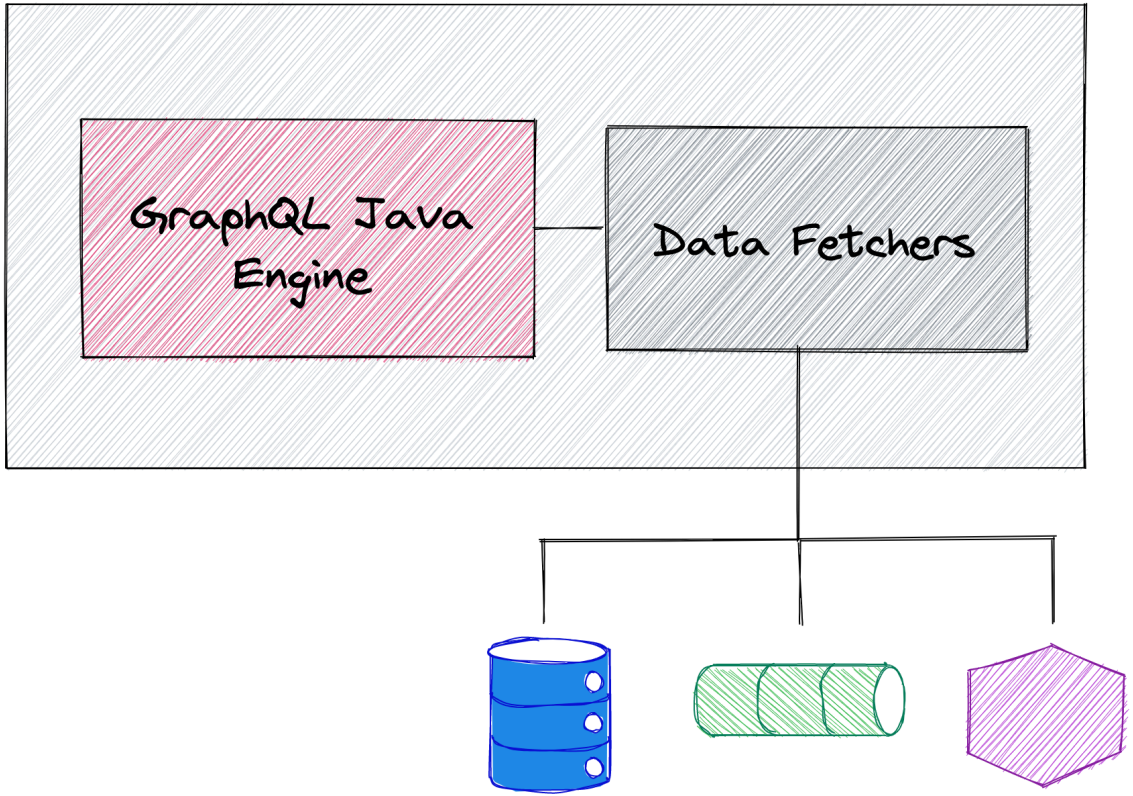
What is Spring for GraphQL

Spring for GraphQL provides support for Spring applications built on GraphQL Java.

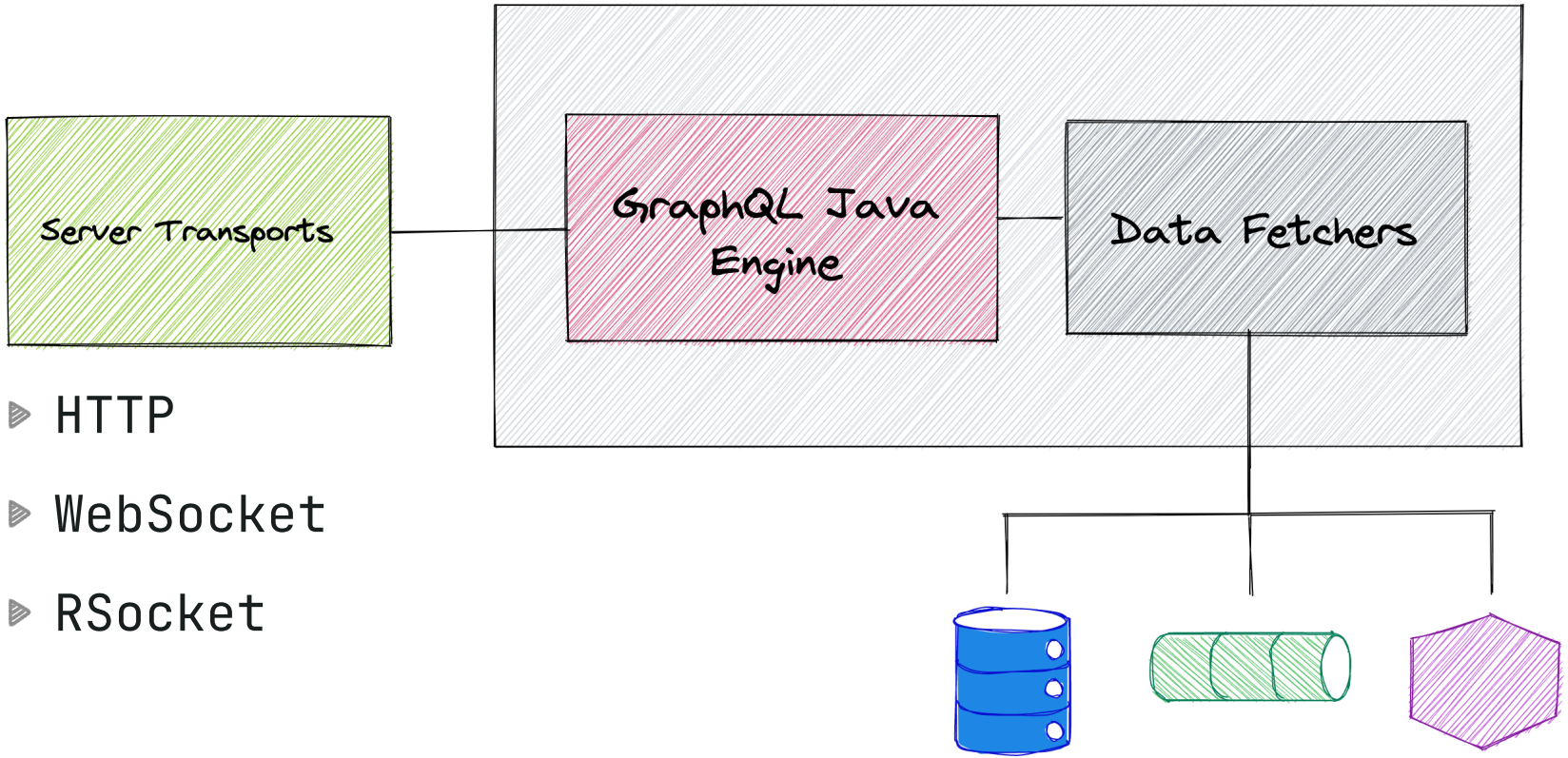
- Joint collaboration between both teams
- GraphQL Java is “limited” on purpose
- Spring Projects - Make complex problems easy
- Requirements
  - JDK8
  - Spring Framework 5.3
  - GraphQL Java 17
  - Spring Data 2021.1.0 or later for QueryDSL or Query by Example
- Spring Boot 2.7+
- Current Version: 1.0.0-M6
- 1.0 GA: 05/17/2022

spring

Spring for GraphQL: Server Transports



Spring for GraphQL: Server Transports



- HTTP
- WebSocket
- RSocket

spring initializr

Project
- Maven Project
- Gradle Project

Language
- Java
- Kotlin
- Groovy

Spring Boot
- 3.0.0 (SNAPSHOT)
- 3.0.0 (M2)
- 2.7.0 (SNAPSHOT)
- 2.7.0 (M3)
- 2.6.7 (SNAPSHOT)
- 2.6.6
- 2.5.13 (SNAPSHOT)
- 2.5.12

Project Metadata

Group	com.example
Artifact	demo
Name	demo
Description	Demo project for Spring Boot
Package name	com.example.demo
Packaging	Jar / War
Java	18 / 17 / 11 / 8

Dependencies

ADD DEPENDENCIES... ⌘ + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring GraphQL WEB

Build GraphQL applications with Spring GraphQL and GraphQL Java.

GENERATE ⌘ + ↵ | EXPLORE CTRL + SPACE | SHARE...

The Spring Boot Starter Graphql (spring-boot-starter-graphql)

- Dependencies
- Autoconfig
- Properties
- Metrics
- GraphiQL UI and Schema Pages

Auto-configured Spring GraphQL Tests

Maven

```xml
<dependencies>
    <dependency>
        <groupId>org.springframework.graphql</groupId>
        <artifactId>spring-graphql-test</artifactId>
        <scope>test</scope>
    </dependency>
    <!-- Unless already present in the compile scope -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-webflux</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Gradle

```gradle
dependencies {
    testImplementation("org.springframework.graphql:spring-graphql-test")
    // Unless already present in the implementation configuration
    testImplementation("org.springframework.boot:spring-boot-starter-webflux")
}
```

GraphQL Slice Test

```
@GraphQlTest(ProductController.class)
class ProductControllerTest {

    @Autowired
    private GraphQlTester graphQlTester;

    @MockBean
    private ProductRepository productRepository;

    private List<Product> products;

    // ...
}
```

- @Controller
- RuntimeWiringConfigurer
- JsonComponent
- Converter
- GenericConverter
- DataFetcherExceptionResolver
- Instrumentation
- GraphQlSourceBuilderCustomizer

spring

Spring for GraphQL: Testing spring-graphql-test

GraphQlTester

- Workflow for testing GraphQL
- Fluent API
- Automatic checks to verify no errors in response
- Use JsonPath to specify data to inspect
- Data Decoding

WebGraphQlTester

- Extension of GraphQL tests over web transports
- Specific HTTP specific inputs
- Uses WebTestClient

```
@Test
void testGetAllProductsQueryReturnsAllProducts() {
    String document = """
        query {
          allProducts {
            id
            title
            desc
          }
        }
    """;

    when(productRepository.findAll()).thenReturn(products);

    graphQlTester.document(document) GraphQlTester.Request<...>
            .execute() GraphQlTester.Response
            .path("allProducts") GraphQlTester.Path
            .entityList(Product.class) GraphQlTester.EntityList<...>
            .hasSize(4);
}
```

Spring for GraphQL Demo

https://github.com/danvega/graphql-store

spring

Thank you

Contact me at dvega@vmware.com

@therealdanvega