

Основи фреймворку Spring Boot. Стартери в Spring Boot. Основи Spring for GraphQL: підхід "schema-first" для побудови GraphQL API

Стельмашук Віталій Володимирович

Львівський національний університет імені Івана Франка
Кафедра інформаційних систем

24 лютого 2025 р.

Outline

Spring Boot

Spring for GraphQL basics

Spring for GraphQL. Annotated controllers

Spring for GraphQL. GraphiQL

Example








Spring Boot overview

- ▶ Spring Boot is a framework for creating stand-alone, production-grade Spring-based applications.
- ▶ It simplifies configuration and dependency management.
- ▶ It provides embedded web servers like Tomcat (by default), Undertow and Jetty.
- ▶ A web application can be started using `java -jar`.
- ▶ Spring Boot Starters
- ▶ Spring Boot Initializr <https://start.spring.io/>

Spring Boot Starters

- ▶ Starters are a set of convenient dependency descriptors that you can include in your application
- ▶ The starters contain a lot of the dependencies that you need to get a project up and running quickly and with a consistent, supported set of managed transitive dependencies
- ▶ All official starters follow a similar naming pattern; `spring-boot-starter-*`, where `*` is a particular type of application
- ▶ Examples:
 - ▶ `spring-boot-starter`: Core starter, including auto-configuration support, logging and YAML
 - ▶ `spring-boot-starter-data-jpa` Starter for using Spring Data JPA with Hibernate
 - ▶ `spring-boot-starter-web` Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container
 - ▶ `spring-boot-starter-graphql` Starter for building GraphQL API

spring-boot-starter-graphql dependencies

- ▼  org.springframework.boot:spring-boot-starter-graphql:3.4.2
 - >  org.springframework.boot:spring-boot-starter-json:3.4.2
 - >  org.springframework.boot:spring-boot-starter:3.4.2
 - ▼  org.springframework.graphql:spring-graphql:1.3.3
 - >  com.graphql-java:graphql-java:22.3
 - >  io.projectreactor:reactor-core:3.7.2
 -  org.springframework:spring-context:6.2.2 (*)

Spring for GraphQL. Server Transports

- ▶ "GraphQL over HTTP" specification draft:
<https://github.com/graphql/graphql-over-http?tab=readme-ov-file>
- ▶ Article that explains best practices how to serve GraphQL over HTTP: <https://graphql.org/learn/serving-over-http/>
- ▶ Single /graphql endpoint
- ▶ Requests must use HTTP POST with "application/json", as content type and GraphQL request details included as JSON in the request body
- ▶ GraphQLWebMvcAutoConfiguration from spring-boot-autoconfigure does the autoconfiguration of this endpoint and handler behind it

Spring for GraphQL. Schema-first. Request execution

- ▶ `ExecutionGraphQLService` is the main Spring abstraction to call GraphQL Java to execute requests
- ▶ The main implementation, `DefaultExecutionGraphQLService`, is configured with a `GraphQLSource` for access to the `graphql.GraphQL` instance to invoke
- ▶ `GraphQLSource` is a contract to expose the `graphql.GraphQL` instance to use that also includes a builder API to build that instance
- ▶ `GraphQLSource.Builder` can be configured with one or more `Resource` instances to be parsed and merged together. That means GraphQL schema files can be loaded from just about any location
- ▶ By default, the Boot starter looks for schema files with extensions `".graphqls"` or `".gqls"` under the location `classpath:graphql/**`, which is typically `src/main/resources/graphql`

Spring for GraphQL. Annotated controllers

Controller components use annotations to declare handler methods with flexible method signatures to fetch the data for specific GraphQL fields.

```
1 @Controller
2 public class GreetingController {
3
4     @QueryMapping
5     public String hello() {
6         return "Hello, world!";
7     }
8
9 }
```


Spring for GraphQL. Annotated controllers

SchemaMapping annotation maps a handler method to a field in the GraphQL schema and declares it to be the DataFetcher for that field. The annotation can specify the parent type name, and the field name:

```
1 @Controller
2 public class BookController {
3
4     @SchemaMapping(typeName="Book", field="author")
5     public Author getAuthor(Book book) {
6         // ...
7     }
8 }
```

Spring for GraphQL. Annotated controllers

```
1  @Controller
2  public class BookController {
3
4      @QueryMapping
5      public Book bookById(@Argument Long id) {
6          // ...
7      }
8
9      @MutationMapping
10     public Book addBook(@Argument BookInput bookInput)
11     ↪ {
12         // ...
13     }
14
15     @SubscriptionMapping
16     public Flux<Book> newPublications() {
17         // ...
18     }
```

Spring for GraphQL. GraphiQL

Spring for GraphQL ships with a stock GraphiQL index.html page that uses static resources hosted on the unpkg.com CDN. Spring Boot applications can easily enable this page with a configuration property:

```
spring.graphql.graphiql.enabled=true
```

Example

- ▶ Spring for GraphQL example

Thank you for your attention!

Questions?