GraphQL Core Concepts

ADHITHI RAVICHANDRAN
SOFTWARE CONSULTANT

@AdhithiRavi www.adhithiravichandran.com

GraphQL is a query language for your API

Queries

GraphiQL

GraphiQL is an in-browser IDE for writing, validating, and testing GraphQL queries.

GitHub’s GraphQL Explorer

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

GitHub’s GraphQL Explorer makes use of your real, live, production data on GitHub

Fields | Arguments | Alias

Fragments | Operation Name | Variables

A GraphQL query is all about asking for specific fields on objects

Arguments

Fields	**Arguments**	Alias
Fragments	Operation Name	Variables

In GraphQL you can pass arguments to fields. Every field and nested object can get its own set of arguments. This gets rid of multiple API fetches.

Alias

Fields

Arguments

Alias

Fragments

Operation Name

Variables

You can’t query for the same field with different arguments. Hence you need aliases. They let you rename the result of a field with anything you want.

Fragments

Fields

Arguments

Alias

Fragments

Operation Name

Variables

Fragments are GraphQL’s reusable units. They let you build sets of fields and then include them in multiple queries.

Fields | Arguments | Alias

Fragments | **Operation Name** | Variables

A meaningful and explicit name for your operation. Think of it like a function name in a programming language.

Variables

Fields

Arguments

Alias

Fragments

Operation Name

Variables

Arguments to fields can be dynamic. GraphQL uses variables to factor dynamic values out of the query, and pass them as a separate dictionary.

Mutations

Mutations

Mutations are used to make changes to the data (Create, update, delete data)

GraphQL assumes side-effects after mutations and changes the dataset after a mutation

While query fields are executed in parallel, mutation fields run in series, one after the other

Types

GraphQL Scalar Types

Int A signed 32-bit integer

Float A signed double-precision floating-point value

String A UTF-8 character sequence

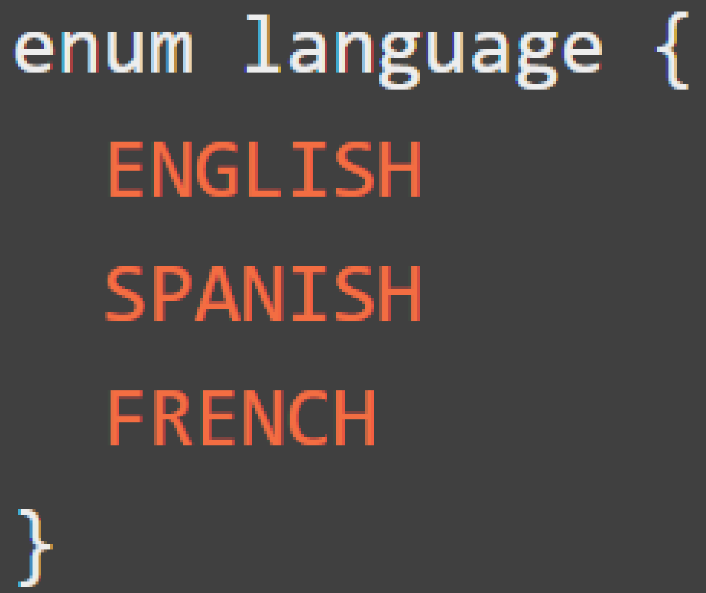
Boolean true or false values

ID Unique identifier. Used to re-fetch an object or as the key for a cache.

GraphQL Types

```
type Author {
  id: ID
  firstName: String
  lastName: String
  rating: Float
  numOfCourses: Int
}
```

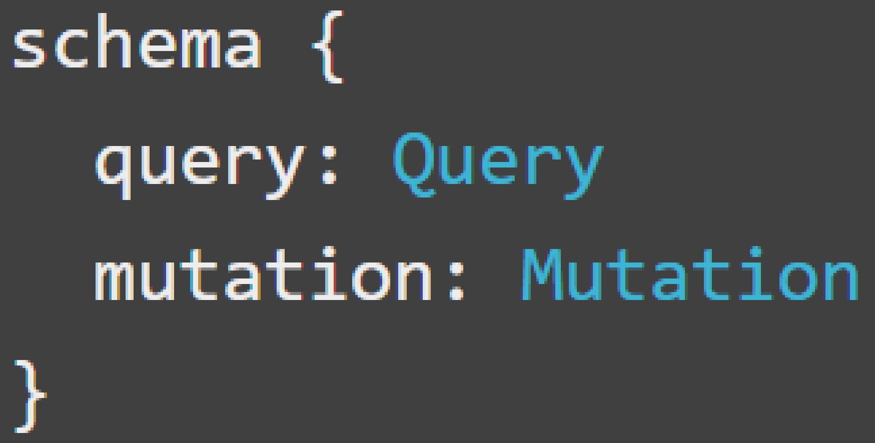
Enumeration Types



```
enum language {
  ENGLISH
  SPANISH
  FRENCH
}
```

- Enums are special scalar types that are restricted to a particular set of allowed values.
- It allows you to validate that any arguments of this type are one of the allowed values.
- This means when we use language in our schema, we expect it to be English, Spanish or French.

Query and Mutation Types



```
schema {
  query: Query
  mutation: Mutation
}
```

Every GraphQL service has a query type. It may or may not have a mutation type. They act as an entry point into the schema.

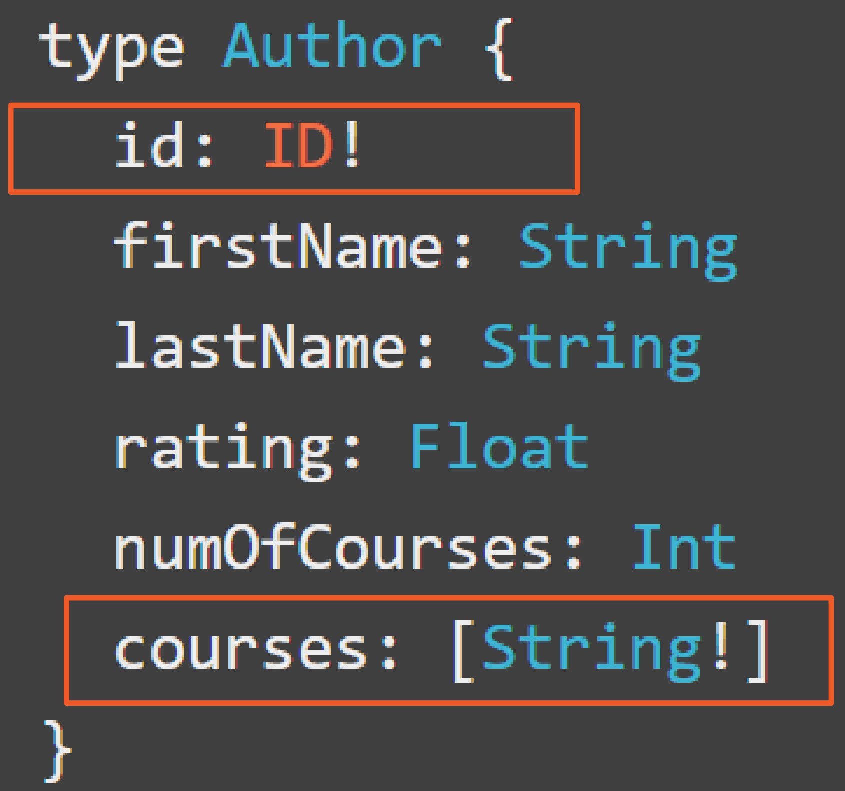
Query and Mutation types are the same as any other GraphQL object type.

Query and Mutation Types

```
type Query {
  author_details: [Author]
}

type Mutation {
  addAuthor(firstName: String, lastName: String): Author
}
```

Non-Nullable Type



```
type Author {
   id: ID!
   firstName: String
   lastName: String
   rating: Float
   numOfCourses: Int
   courses: [String!]
}
```

- By default each of the core scalar types can be set to null.
- To override this default behavior and ensure that a field cannot be null, the ! Is used.
- Here the Author object will always have an id. Since it is declared as a non-nullable field.
- You can even have a list of Non-Null items. (List of non-null courses)

Summary

GraphQL Core Concepts

- Schemas and Types
- Queries
- Mutations

Next Module: Why GraphQL?