Reseach paper

GraphQL

Patrick Richter

Fontys University of Applied Sciences

Venlo, January 14, 2019

Summary

This paper serves as the research paper for the Sofa project related to the topic GraphQL compared to REST. Students of the Software Engineering and Business Informatics course of Fontys University of Applied Sciences have to take part in a project known as the module *Software Factory* (SOFA). This project contains the development of a software product for a client that assigned the project to be worked on by students of the university. The customer connected.Football is using this technology in their application with success. This research paper summarizes the pros and cons of the GraphQL API which is put on top of REST. GraphQL is a framework built by Facebook in 2012 and first released to the public at 2015. It is built on top of Rest architecture to help companies fasten their software production life cycle and serves the fast changing data exchange through the internet in the 21st century.

Contents

List of Figures

List of Abbreviations

REST Representational State Transfer

API Application Program Interface

IOT Internet of things

1 Introduction

Technology is advancing at an ever increasing pace, especially in the realm of software development. Every day giant data is transferred and send through the internet. It is hard to take track of the data and the technology that is transferring the data. RESTFUL is a old convention that is used to pass, change and retrieve data from databases by many programmers. To keep up with the demand, Facebook developed an API called GraphQL[1] to serve the purpose to keep up with the growing speed of software development cycles. GraphQL, as its core enables declarative data fetching which grants users the ability to retrieve the exact data they want and need from the database. A GraphQL server only exposes a single endpoint instead of multiple ones when using REST and responds with data a user precisely asks for.

GraphQL defines it self as a query language for APIs. That's being said, most applications and devices whether it be a IOT or smart phone has to fetch data from a server that is storing relevant data to the user. With the help of an API an interface is being provided to retrieve or change data. People mistakenly believe GraphQL is a database technology, which is not. Some people even see GraphQL as an replacement for REST. It's a framework for RESTFUL services that can be implemented in multiple different programming languages or frameworks where an API is being used.

[1] `https://graphql.org`

2 Problem domain

An improved alternative to REST is to expose data from a server. REST has proven to be a sufficient way of doing so. When REST has been designed and developed. Back in the days web applications had been fairly simple and general development speed wasn't nearly that fast as where it is today, having multiple releases a day with continuous integration life cycles. Over the last couple of years, the API landscape has changed tremendous so in context, there are three main points that need to be considered in terms of API design:

2.1 Increasing data

More data is being consumed by mobile devices. GraphQL reduces the amount of data that is being transferred over the network. Facebook invented GraphQL to deal with low-powered devices and sloppy networks

2.2 New technology

Endless front end frameworks and upcoming platform Increasing growth of landscapes and frameworks that handle client application's makes it nearly impossible to have only one API that fits all requirements. Through the help of GraphQL, clients can use declarative fetching to only retrieve data that is needed.

2.3 Increased development cycles

The increasing feature development. These days, many companies make use of continuous deployment, where weekly product updates and rapid iterations are the standard of software development life-cycles. Development practices and product iterations are influenced by the way a server spreads data. Often the developer has to change requirements to fit the needs of the REST architecture which decreases development time.

3 Core concepts

3.1 Schema Definition Language

GraphQL has a syntax for defining types as well as executing queries and mutations called "Schema Definition Language (SDL)"[2]. GraphQL has a easy to understand syntax, it could be compared to Python in some ways. In GraphQL, to define a type called Animal, following syntax is used:

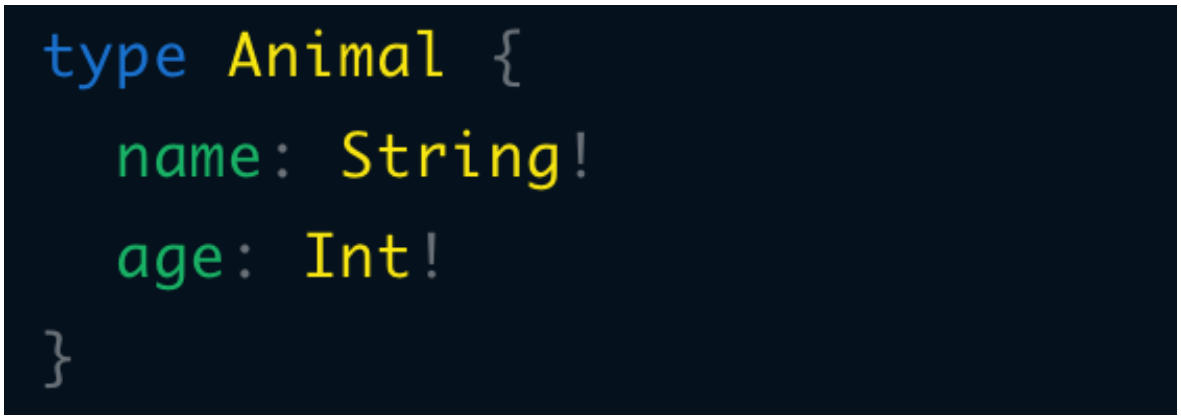

Figure 1: Simple type definition in GraphQL

This example expresses two fields, name and age with the corresponding type. While the exclamation mark expresses that the field is required, it is not necessary to use it. to be able to fetch data with the help of queries, GraphQL APIs exposes only a single endpoint compared to REST where multiple endpoints are being exposed. This can be achieved, because the data structure that is returned is not fixed, it is totally flexible and gives the client the power to decide which data is going to be returned from the server.

[2] https://www.howtographql.com/basics/2-core-concepts/

3.2 Mutation

To change data on the server GraphQL works with mutations.[3] They do have the same structure as the previous mentioned queries, the key difference in the first place is the keyword mutation instead of queries. There are three main mutations provided:
1. delete data
2. create data
3. update data

Following example shows how to change data:

Figure 2: Simple mutation definition in GraphQL

[3] https://graphql.github.io/learn/queries/#mutations

3.3 Subscribing to events

Many applications have a real time connection between client and server, it is important to allow users to be notified immediately, if a change has happened. GraphQL solves this with so called subscriptions, and establishes a steady connection, if a client subscribes to an event. The user from now on will be informed by the server, if the subscribed event actually happens. Subscription are different, they do follow a cycle of "request-and-response" which are presented in a steam of data that is being sent to the client. Following an example of subscription:

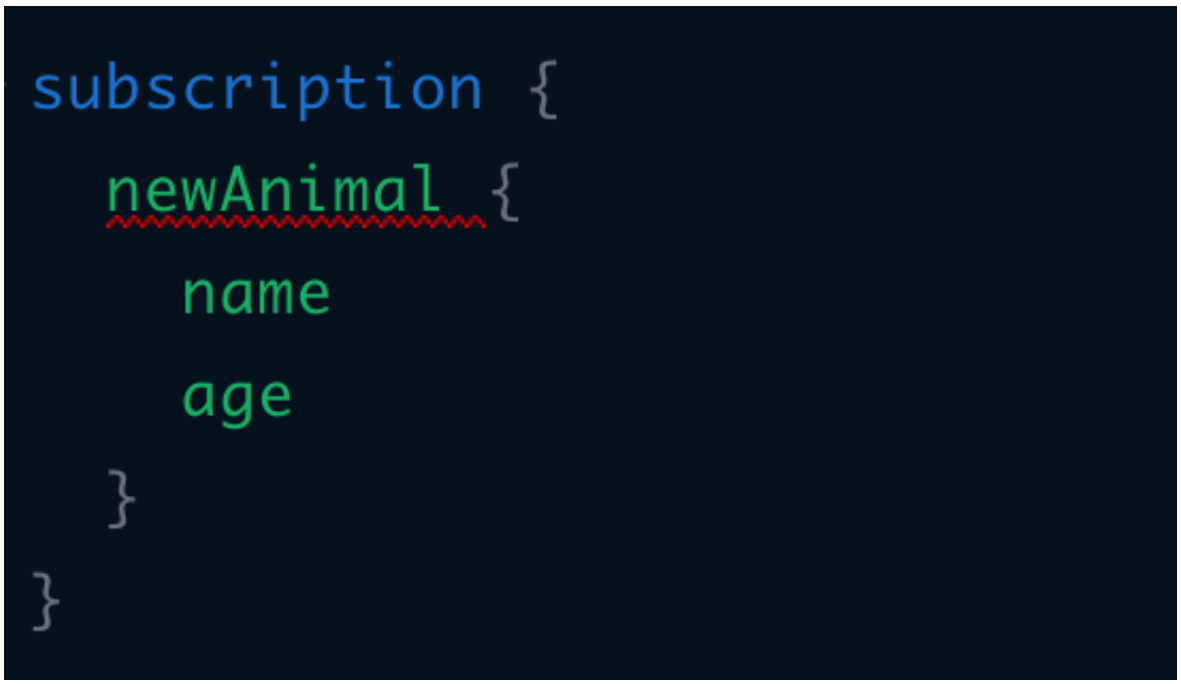

Figure 3: Simple subscription definition in GraphQL

4 Conclusion

As right now, December,2018 GraphQL is an interesting technology which addresses common and well known problems of the REST architecture, might it be lacking power of mobile devices or the excessive data that is being transferred over the network. GraphQL's architecture is designed to improve development speed as well as sloppy network communication and weak mobile CPU-devices. Currently REST is a widely used technique that is used in a lot of stacks and it is hard to tell if GraphQL will replace it anytime soon. However, GraphQL shows that it has the potential to disrupt the sector due to its promising features and the open variety in terms of applicable use cases in terms of programming languages, frameworks and official support. Only time will tell if it will replace REST. Upcoming events that are dedicated to GraphQl e.g. "GraphQL Summit"[4] which will take place from the 7-8th of May 2019 are well visited and bring improvements to the language. Since Facebook invented and supports this technology, it is to be assumed that it is not going to vanish in the near future.

[4]https://summit.graphql.com/