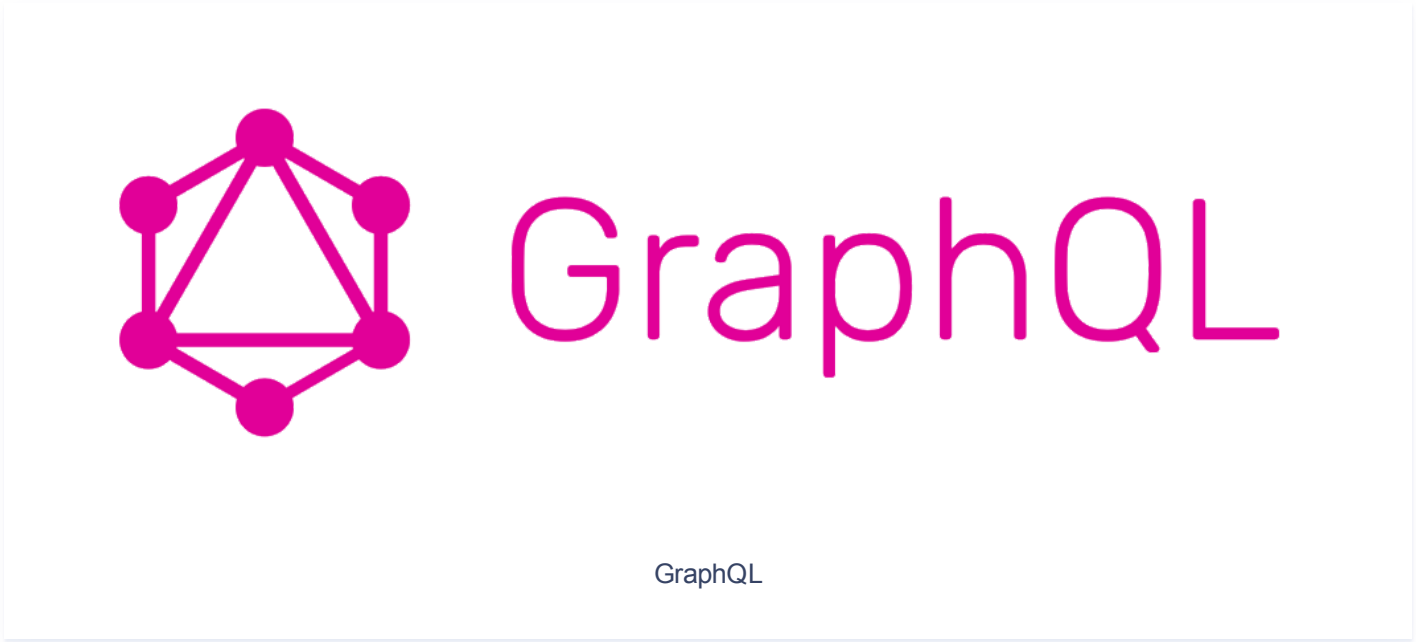
Руководство по GraphQL

В статье мы рассмотрим, что такое GraphQL, как выглядят запросы и зачем нужна схема GraphQL.

Что такое GraphQL

GraphQL — это язык запросов для API-интерфейсов и среда, в которой они выполняются. С помощью GraphQL можно получить данные из API и передать их в приложение (от сервера к клиенту). Официальная документация GraphQL есть только на английском языке, на русский язык пока еще не переведена.



GraphQL

GraphQL разработали в 2012 году как альтернативу REST. Напомним, что REST — это архитектурный стиль, который описывает правила взаимодействия между клиентом и сервером. При этом используются HTTP-запросы, стандарт построения URL, форматы данных JSON и XML. REST API выдает ответ на запрос в формате JSON. Затем ответ нужно проанализировать, выбрать конкретные данные и извлечь их. GraphQL же позволяет передать в приложение сразу нужные данные **за один запрос**, даже если они находятся в нескольких источниках. Благодаря этому технология извлечения данных GraphQL удобнее и практичнее, чем REST.

Итак, перечислим основные преимущества GraphQL:

- Не нужно создавать несколько REST-запросов. Чтобы извлечь данные, достаточно ввести один запрос.
- Не привязан к конкретной базе данных или механизму хранения.
- Используется целая система типов данных.

GraphQL API построен на двух основных блоках: запросах (queries) и схеме (schema).

GraphQL: запросы

Сначала запросы в GraphQL (**query**) отправляют на сервер. Затем они возвращаются как ответ клиенту (приложению) в формате JSON. Неважно, откуда поступают данные, их можно запросить конкретной командой.

Простейший запрос GraphQL может выглядеть так:

```
query {


  friends {
    name
    others {
      name
    }
  }
}
```

Такой запрос выдаст следующий ответ в формате JSON:

```
{

   "data": {
     "friends": {
       "name": "Alex",
       "others": [
         {
           "name": "Mike"
         },
         {
           "name": "Jason"
         },
         {
           "name": "Jack"
         }
       ]
     }
   }
 }
```

Как видно выше, ответ имеет такую же форму, как и запрос. Поэтому в GraphQL ответ всегда ожидаем.

Самый распространенный способ работы с API GraphQL — графический инструмент **GraphiQL**. Когда вы подключаете GraphiQL к конечной точке (endpoint) GraphQL, он запрашивает у сервера свою **схему** GraphQL и предоставляет вам пользовательский интерфейс для просмотра и тестирования запросов. GraphiQL

отправляет строку запроса GraphQL на сервер в заданной конечной точке с заголовками HTTP. Затем сервер отправляет ответ, который получает пользователь.

Подробнее о схемах ниже.

GraphQL: схема

В целом, язык запросов GraphQL — это выбор полей в объектах. Так, в примере запроса ниже мы:

выбираем поле friend,

для объекта, который возвращён в friend, выбираем поля name и address:

```
query {
  friend {
    name
    address
  }
}
```

Запрос GraphQL совпадает с результатом, и мы можем предсказать, какой получим ответ, без знаний о сервере. Однако удобнее иметь точное описание данных для запроса: какие поля можно выбрать, и какие типы объектов будут возвращены в ответе. Как раз для этого и нужна схема GraphQL.

Любой сервис GraphQL определяет набор типов данных, которые можно запросить. Каждый ответ на запрос проверяется и выполняется только в соответствии со схемой.

Самые простые компоненты схемы GraphQL — **типы объектов**. Каждый тип содержит вид объекта и поля для этого объекта. В GraphQL schema language мы можем отобразить это следующим образом:

```
type Character {
  name: String
  address: [City]
}
```

Character — это тип объекта GraphQL: тип и некоторые поля.

name и **address** — это поля в Character. Эти поля могут появиться в ответе на запрос GraphQL.

String — тип данных (в данном случае это строка). Также здесь могут быть и другие типы — Int (целое число), Float (число с плавающей точкой), Boolean (true или false), ID (уникальный идентификатор).

[City] — массив объектов City.