Django Essential

Graphql django api (часть 1)

Django Essential

Introduction

Лазорык Михаил

Software developer, 3 года опыта

mykhailo.lazoryk

mykhailo-lazoryk

Тема урока

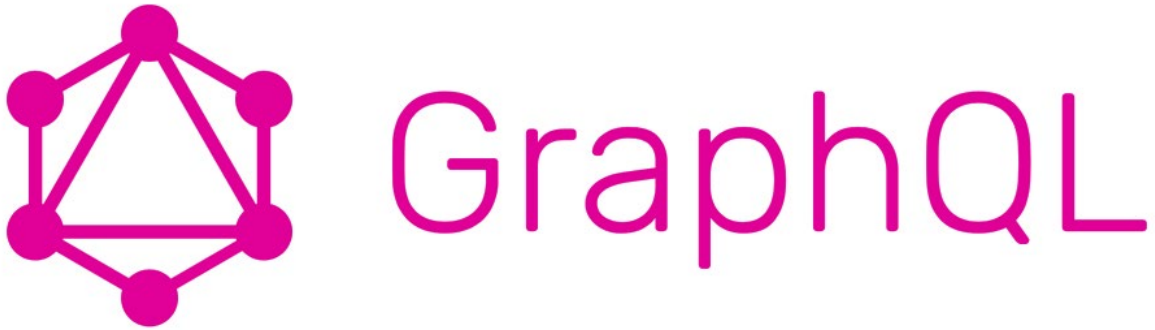
Graphql django api (часть 1)

Graphql django api (часть 1)

1. Что такое GraphQL и примеры использования
2. Создание простого запроса в онлайн IDE
3. Рассмотрение схем
4. Распознаватели
5. Мутации
6. Сравнение Graphql с REST API

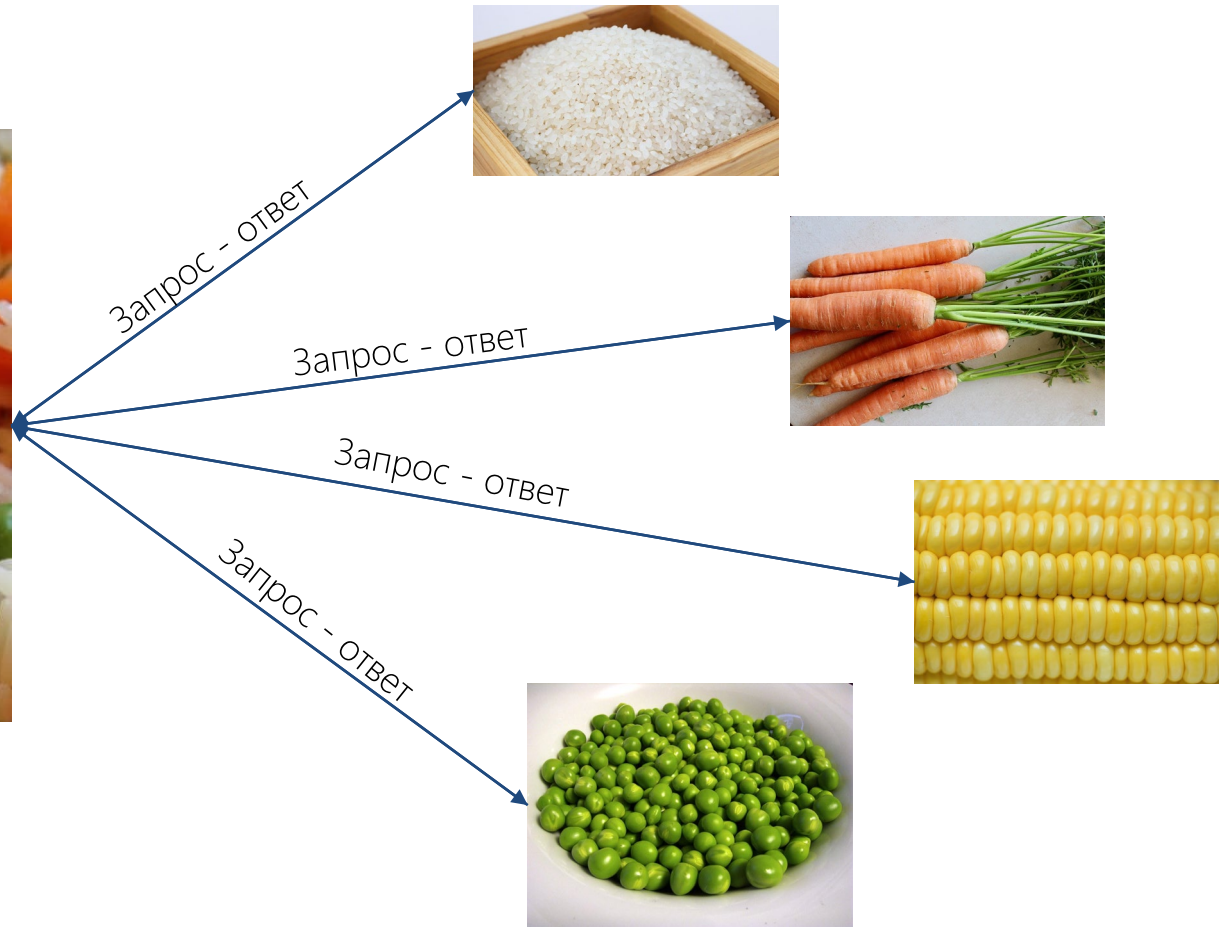
Что такое GraphQL и примеры использования

GraphQL - язык запросов с открытым кодом. Используется для создания запросов и для и их обработки. Язык создан в Facebook. Написан на Java, JavaScript, Ruby и Scala.



Что такое GraphQL и примеры использования

REST



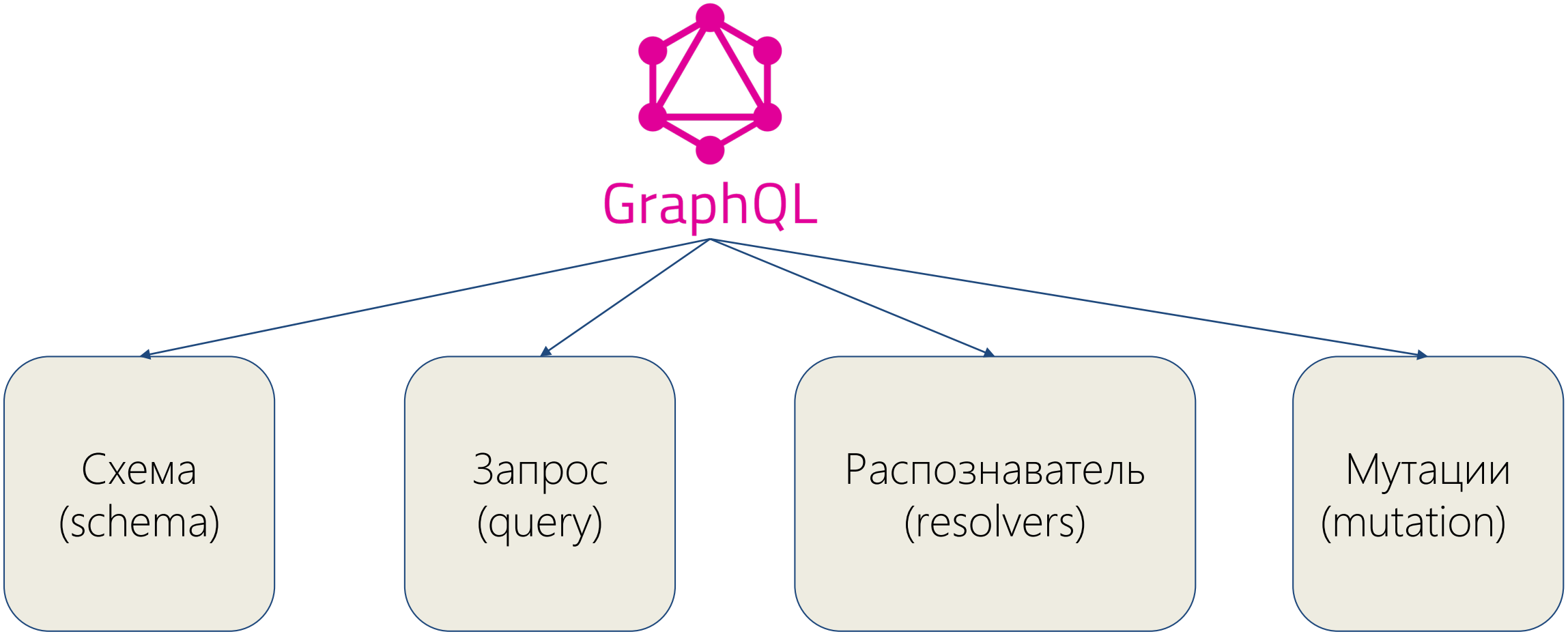
Что такое GraphQL и примеры использования

GraphQL

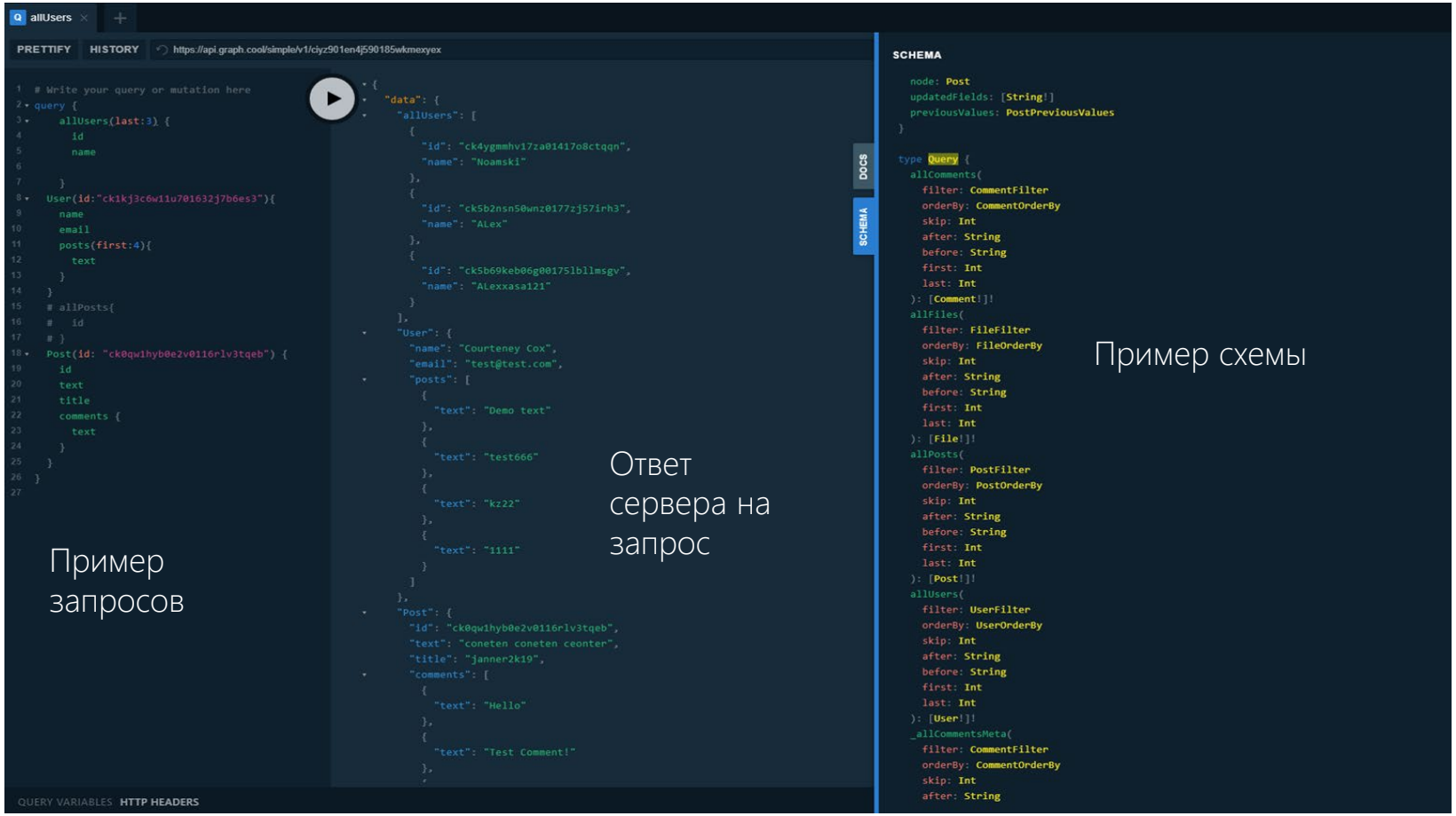
Запрос - ответ

GraphQL

Что такое GraphQL и примеры использования



Создание простого запроса в онлайн IDE



https://api.graph.cool/simple/v1/ciyz901en4j590185wkmexyex

Рассмотрение схем

- Схема (schema) - корень любого сервера, который реализует GraphQL. Это место, где хранится информация о типах данных, которые будут использоваться и где выполняются queries (запросы).
- Создание схем происходит с помощью внутреннего языка SDL (Schema Definition Language).
- В схемах можно использовать различные кастомные типы.
- Поля могут быть Int, Float, String, Boolean, ID, List of Object Types, или Custom Objects Types.
- Типы могут иметь поля.
- Можно использовать фильтры.

```
type Post implements Node {
  comments(
    filter: CommentFilter
    orderBy: CommentOrderBy
    skip: Int
    after: String
    before: String
    first: Int
    last: Int
  ): [Comment!]
  createdAt: DateTime!
  id: ID!
  text: String!
  title: String!
  updatedAt: DateTime!
  user(filter: UserFilter): User
  _commentsMeta(
    filter: CommentFilter
    orderBy: CommentOrderBy
    skip: Int
    after: String
    before: String
    first: Int
    last: Int
  ): _QueryMeta!
}
```

Распознаватели

- Распознаватель нужен для того, чтобы GraphQL понимал где ему взять данные для ответа на запрос.

```
Comment(id: ID): Comment
File(id: ID, secret: String, url: String): File
Post(id: ID): Post
User(email: String, id: ID): User
user: User
node(id: ID!): Node
```

```
type Post implements Node {
  comments(
    filter: CommentFilter
    orderBy: CommentOrderBy
    skip: Int
    after: String
    before: String
    first: Int
    last: Int
  ): [Comment!]
  createdAt: DateTime!
  id: ID!
  text: String!
  title: String!
  updatedAt: DateTime!
  user(filter: UserFilter): User
  _commentsMeta(
    filter: CommentFilter
    orderBy: CommentOrderBy
    skip: Int
    after: String
    before: String
    first: Int
    last: Int
  ): _QueryMeta!
}
```

Мутации

- Распознаватель, который изменяет что то в базе данных называется мутацией.
- Мутации служат для модификации полей в базе, то есть запись, обновление, удаление.

```
type Mutation {
  createComment(
    text: String!
    postId: ID
    post: CommentpostPost
    userId: ID
  ): Comment
  createFile(name: String!): File
  createPost(
    text: String!
    title: String!
    userId: ID
    commentsIds: [ID!]
    comments: [PostcommentsComment!]
  ): Post
  updateComment(
    id: ID!
    text: String
    postId: ID
    post: CommentpostPost
    userId: ID
  ): Comment
  updateFile(id: ID!, name: String): File
  updatePost(
    id: ID!
    text: String
    title: String
    userId: ID
    commentsIds: [ID!]
    comments: [PostcommentsComment!]
  ): Post
  updateUser(
    id: ID!
    name: String
    commentsIds: [ID!]
    comments: [UsercommentsComment!]
    postsIds: [ID!]
    posts: [UserpostsPost!]
  ): User
  updateOrCreateComment(update: UpdateComment!, create: CreateComment!): Comment
  updateOrCreateFile(update: UpdateFile!, create: CreateFile!): File
```

Сравнение Graphql с REST API

GraphQL

- вместо работы с жестко установленными конечными точками (endpoints), можно получить именно те данные, которые нужны.
- описание ресурса не связано со способом его получение, так как есть прослойка GraphQL которая делает всю грязную работу.
- GraphQL не использует url для идентификации того, что доступно в API, вместо этого используется схема.
- GraphQL использует множество функций.
- Для записи меняется слово в запросе

REST API

- Вызывает одну функцию обработчика
- Запись данных определяется как HTTP метод (POST)

Проверка знаний

TestProvider.com

Проверьте как Вы усвоили данный материал на TestProvider.com

TestProvider – это online сервис проверки знаний по информационным технологиям. С его помощью Вы можете оценить Ваш уровень и выявить слабые места. Он будет полезен как в процессе изучения технологии, так и для общей оценки знаний IT специалиста.

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.

Спасибо за внимание! До новых встреч!

Лазорык Михаил
Software developer

Информационный видеосервис для разработчиков программного обеспечения

TestProvider Pearson CyberBionic systematics Microsoft Partner Windows Azure EBA