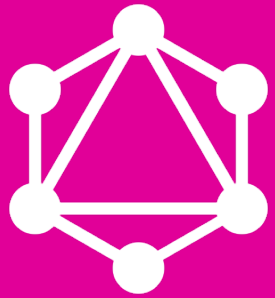


GraphQL 101

An introduction to the world of GraphQL

Francesca Guiducci

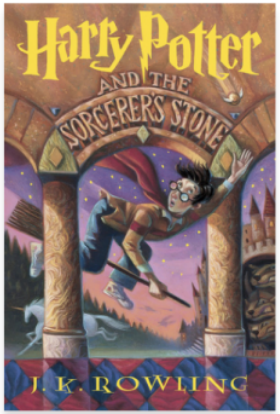
Backend Software Engineer

@engfragui

REST

REST API

Home My Books Browse Community Search books



Read

My rating:

Preview Listen

Harry Potter and the Sorcerer's Stone (Harry Potter #1)

by J.K. Rowling, Mary GrandPré (Illustrator), Gianni Pannofino (Narrator)

4.45 · Rating details · 4,938,695 Ratings · 78,193 Reviews

Harry Potter's life is miserable. His parents are dead and he's stuck with his heartless relatives, who force him to live in a tiny closet under the stairs. But his fortune changes when he receives a letter that tells him the truth about himself: he's a wizard. A mysterious visitor rescues him from his relatives and takes him to his new home, Hogwarts School of Witchcraft ...more

GET A COPY

Kindle Unlimited $0.00 | Amazon | Stores | Libraries

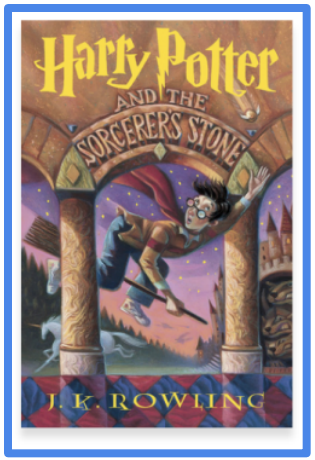
Or buy for $8.99

ABOUT J.K. ROWLING

See also: Robert Galbraith

Although she writes under the pen name **J.K. Rowling**, pronounced like *rolling*, her name when her first *Harry Potter* book was published was simply **Joanne Rowling**. Anticipating that the target audience of young boys might not want to read a book written by a woman, her publishers demanded that she use two initials, rather than her full name. As she had no middle name, she ch ...more

More about J.K. Rowling...



Harry Potter and the Sorcerer's Stone (Harry Potter #1)

by J.K. Rowling, Mary GrandPré (Illustrator), Gianni Pannofino (Narrator)

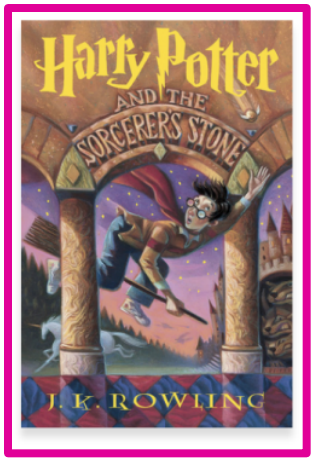
Harry Potter's life is miserable. His parents are dead and he's stuck with his heartless relatives, who force him to live in a tiny closet under the stairs. But his fortune changes when he receives a letter that tells him the truth about himself: he's a wizard. A mysterious visitor rescues him from his relatives and takes him to his new home, Hogwarts School of Witchcraft ...more

ABOUT J.K. ROWLING

See also: Robert Galbraith
Although she writes under the pen name **J.K. Rowling**, pronounced like *rolling*, her name when her first *Harry Potter* book was published was simply **Joanne Rowling**. Anticipating that the target audience of young boys might not want to read a book written by a woman, her publishers demanded that she use two initials, rather than her full name. As she had no middle name, she ch ...more

`/book/12345`
`/bookpic/12345`
`/author/6666`
`/recommendedbooks/12345`
`/bookquotes/12345`
`/bookreviews/12345`

1. Multiple calls
2. Over-fetching



Harry Potter and the Sorcerer's Stone (Harry Potter #1)

by J.K. Rowling, Mary GrandPré (Illustrator), Gianni Pannofino (Narrator)

Harry Potter's life is miserable. His parents are dead and he's stuck with his heartless relatives, who force him to live in a tiny closet under the stairs. But his fortune changes when he receives a letter that tells him the truth about himself: he's a wizard. A mysterious visitor rescues him from his relatives and takes him to his new home, Hogwarts School of Witchcraft ...more

ABOUT J.K. ROWLING

See also: Robert Galbraith
Although she writes under the pen name **J.K. Rowling**, pronounced like *rolling*, her name when her first *Harry Potter* book was published was simply **Joanne Rowling**. Anticipating that the target audience of young boys might not want to read a book written by a woman, her publishers demanded that she use two initials, rather than her full name. As she had no middle name, she ch ...more

`/bookpage/12345`

`/bookpagewithoutdescription/12345`

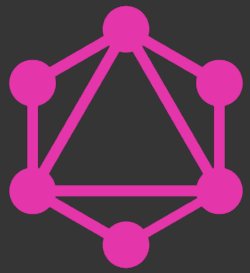
`/bookpagewithoutdescriptionbutwithtwopictures/12345`

REST

```
/book/12345
/bookpic/12345
/author/6666
/bookpage/12345
```

- Client calls many APIs/endpoints
- No easy way to specify which fields
- Tempted to build custom endpoints
- Need to write documentation

We would really need an alternative to REST that gets rid of all those issues



GraphQL

Query language to select exactly what you want

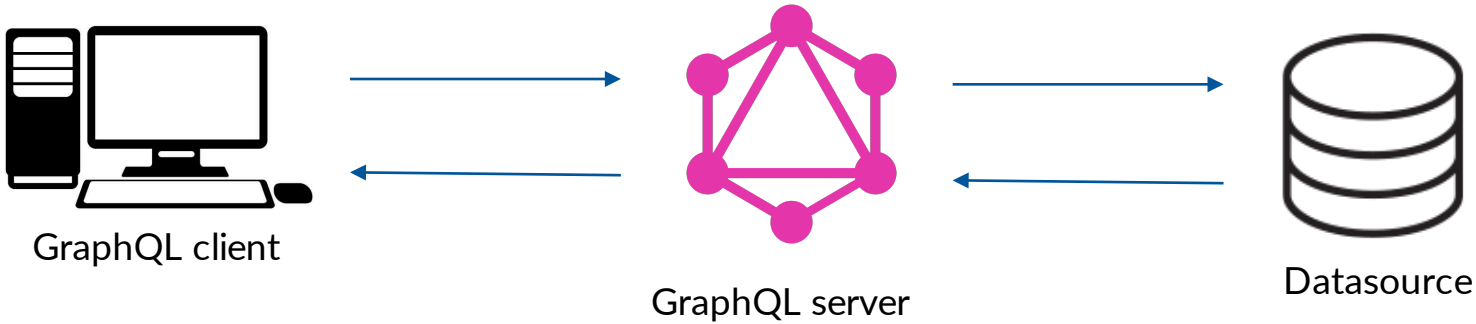
“One endpoint to rule them all”

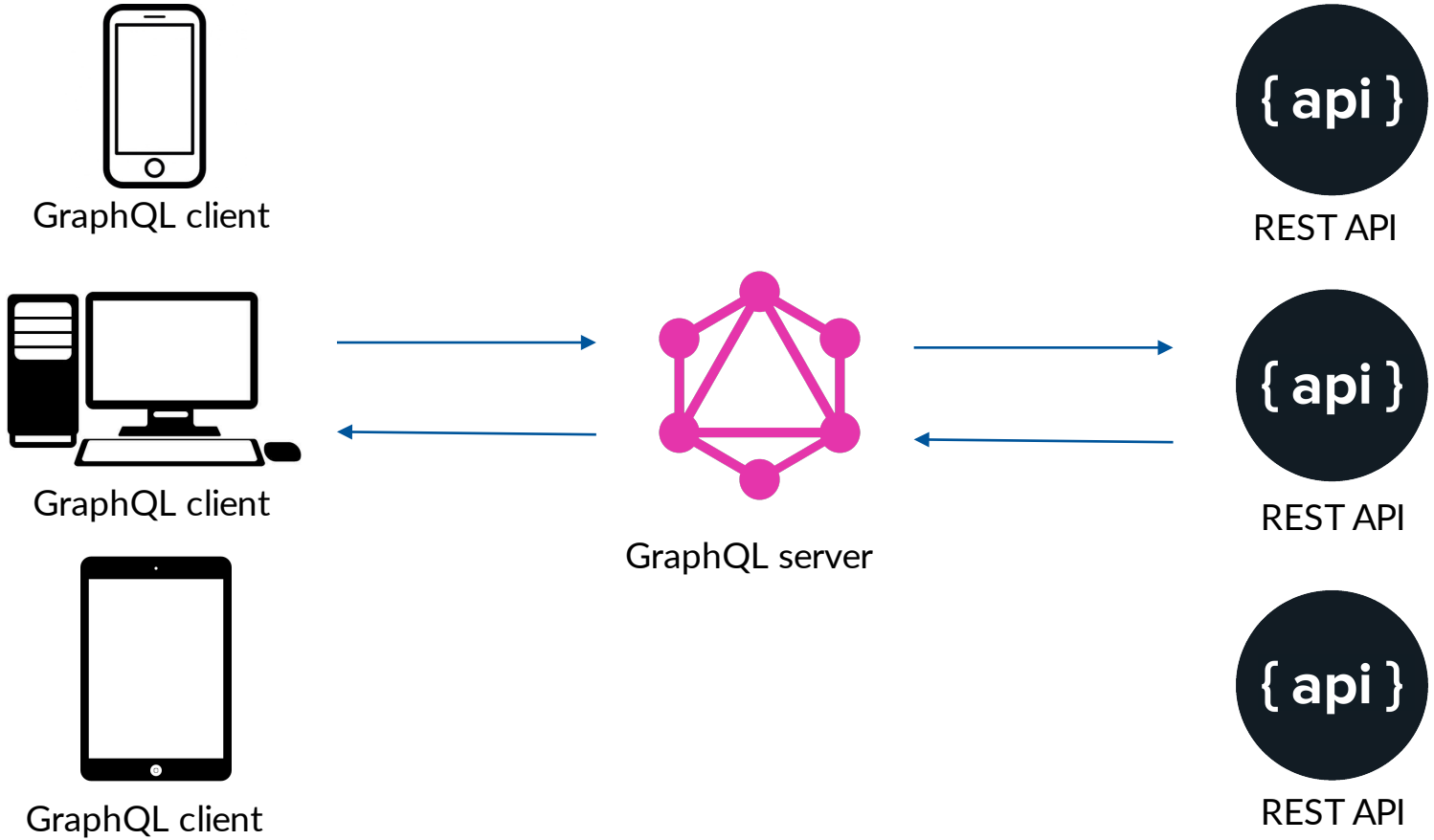
GraphQL

```
query {
  book(id: 1234){
    title
    author {
      name
    }
  }
}
```

- Alternative to REST
- Client queries only one endpoint
- Client can specify which fields
- Documentation comes “for free”

How will my architecture look like?


GraphQL client
GraphQL server
Datasource


GraphQL client
GraphQL client
GraphQL client
GraphQL server
{ api }
REST API
{ api }
REST API
{ api }
REST API

GraphQL Schema Definition

```
schema {
    query: Query
}

type Query {
    book(isbn: String!): Book
}

type Book {
    id: String
    isbn: String
    title: String
    author: Author
    review: Review
}
```

```
type Author {
    id: String
    name: String
}

type Review {
    id: String
    stars: Int
    content: String
}
```

GraphQL Operations

- *Queries*
- *Mutations*
- *Subscriptions*

Queries

request

```
query {
  book(id: "88888") {
    id
    isbn
    title
    author {
      name
    }
  }
}
```

response

```
"data": {
  "book": {
    "id": "88888",
    "isbn": "0618346252",
    "title": "The Fellowship of the King",
    "author": {
      "name": "J.R.R. Tolkien"
    }
  }
}
```

Mutations

request

```
mutation {
  updateBook (
    id: "88888",
    title: "The Fellowship of the Ring"
  ) {
    id
    title
  }
}
```

response

```
"data": {
  "book": {
    "id": "88888",
    "title": "The Fellowship of the Ring"
  }
}
```

Subscriptions

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 57
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 52
  }
  ]
}
```

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 57
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 53
  }
  ]
}
```

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 57
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 54
  }
  ]
}
```

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 57
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 55
  }
  ]
}
```

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 57
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 56
  }
  ]
}
```

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 58
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 56
  }
  ]
}
```

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 58
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 57
  }
  ]
}
```

request

```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 58
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 58
  }
  ]
}
```

request

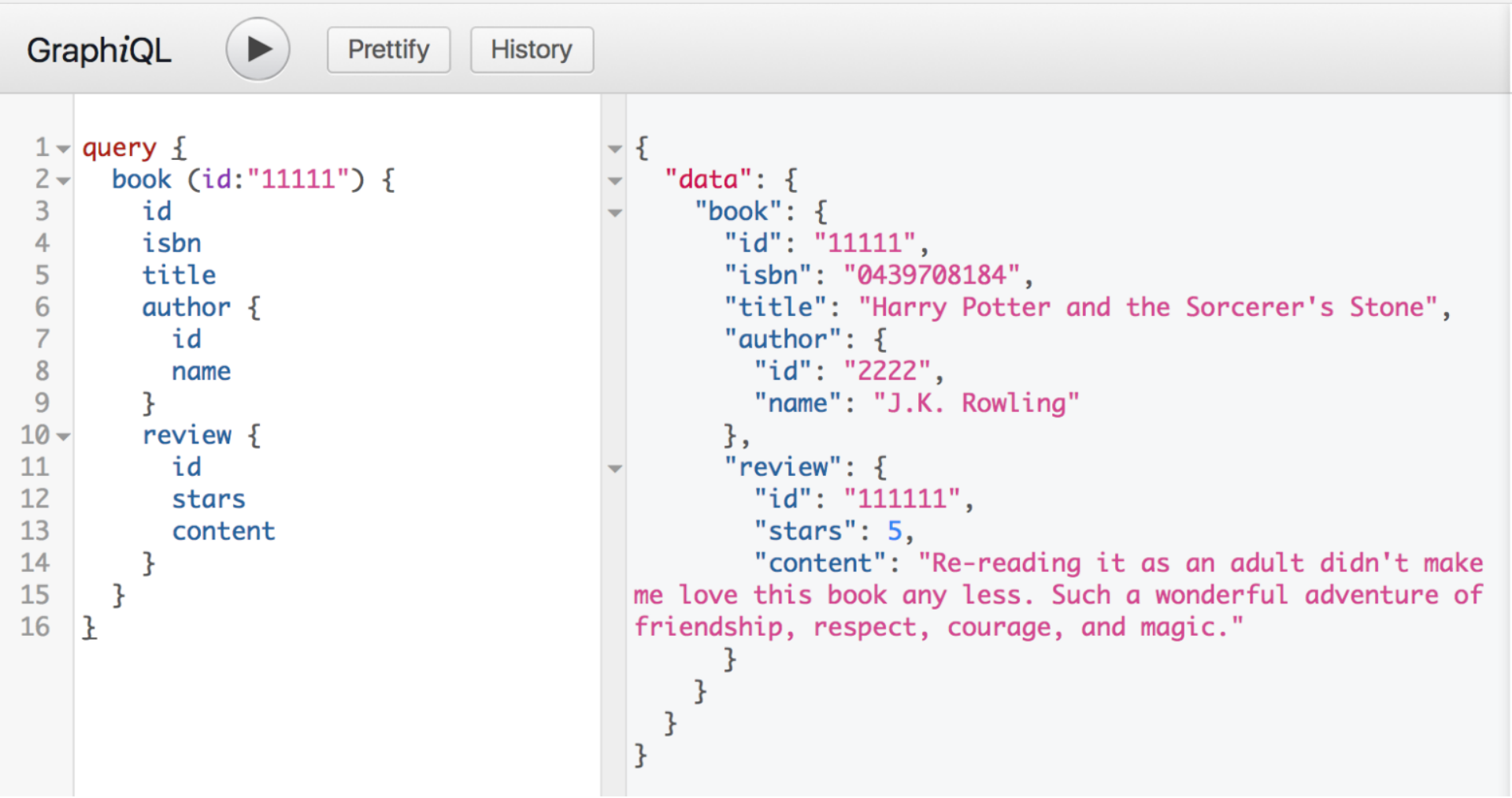
```
subscription {
  likesUpdated {
    id
    title
    likes
  }
}
```

response

```
"data": {
  "likesUpdated": [
  {
    "id": "88888",
    "title": "The Fellowship of the Ring"
    "likes": 58
  },
  {
    "id": "99999",
    "title": "A Game of Thrones"
    "likes": 59
  }
  ]
}
```

Demo

github.com/engfragui/simple-graphql-server


GraphiQL
Prettify
History
query {
book (id:"11111") {
id
isbn
title
author {
id
name
}
review {
id
stars
content
}
}
}
{
"data": {
"book": {
"id": "11111",
"isbn": "0439708184",
"title": "Harry Potter and the Sorcerer's Stone",
"author": {
"id": "2222",
"name": "J.K. Rowling"
},
"review": {
"id": "111111",
"stars": 5,
"content": "Re-reading it as an adult didn't make me love this book any less. Such a wonderful adventure of friendship, respect, courage, and magic."
}
}
}
}

Drawbacks

and possible solutions

Many queries to data layer	solve by →	Batching
No “out of the box” HTTP caching	solve by →	Normalized cache
Malicious queries	solve by →	Timeout or limit query depth

GraphQL Recap

- *Query language alternative to REST*
- *Operations* | *Queries, mutations, subscriptions*
- *Drawbacks and possible solutions*

Resources

- GraphQL official documentation
- Zero to GraphQL in 30 Minutes
- From REST to GraphQL
- Beyond REST: Coursera's Journey to GraphQL
- Wrapping a REST API in GraphQL
- Under the Hood of The Times Website
- REST versus GraphQL
- Mental models for teaching GraphQL
- Making mistakes with GraphQL
- GraphQL for iOS Developers
- GraphQL Talks

Thank you

Francesca Guiducci

Backend Software Engineer

@engfragui