


REST vs GraphQL

by Pawan Kumar
Software Engineer at SquareBoat




The REST Way

Topic Page



PHP Tutorials and Courses


Learn PHP online from the best PHP tutorials & courses recommended by the programming community.

 Share
  Tweet
  Share


Topic

Top tutorials


Upvotes ↓ | Recent


53


[PHP The Right Way](#) (phptherightway.com)

 Save Submitted by Khairul [Free](#)

Tutorial


18

[PHP Tutorial](#) (nusphere.com)

 Save Submitted by Sam [Free](#)

Tutorial


12

 [The complete PHP 5 tutorial](#) (php5-tutorial.com)

 Save Submitted by Sam [Free](#)

Tutorial


0

 [PHP y MYSQL: El Curso Completo, Practico y Desc](#)

 Save Submitted by Jose [Paid](#) [Video](#) [Beginner](#)

Tutorial


8

[PHP for Beginners](#) (udemy.com)

 Save Submitted by Howard [Paid](#) [Video](#) [Beginner](#)

Tutorial

CHOOSE YOUR COURSE

Type of course

☐ Free (10)

☐ Paid (6)

☐ Video (9)

☐ Book (1)

☐ Exercises/Practice-programs (2)

☐ Español (1)

Level

☐ Beginner (5)

Version

☐ PHP 7 (1)

☐ PHP 5 (1)

REST-ful API

GET - /topics/php

GET - /topics/php/tutorials

GET - /topics/php/related_tutorials

or

GET - /topics/php?include=tutorials,related_tutorials

or

GET - /topics/php?include=tutorials(name,description,vote_count)

REST-ful API Issues

- Response is Implicit
- Too many Round trips
- No way to get limited fields
- No way to manipulate nested resources

GraphQL Way

What is GraphQL ?

- Query Language
- Spec

What GraphQL is not?

- A Database
- Library
- Language Specific

The Query

```
{
  users {
    id
    name
  }
}
```

```
{
  "users": [
    {
      "id": 1,
      "name": "Swapnil Banga"
    },
    {
      "id": 2,
      "name": "Pawan Kumar"
    },
    {
      "id": 3,
      "name": "Rajat Patel"
    }
  ]
}
```

What you asked is what you get

GRAPHIQL

History

×

createUser

createUser

query { users { id firstName...

mutation { createUser(input:...

query { users { id firstName...

mutation { createUser(input:...

mutation { createUser(input:...

mutation { createUser(input:...

query { users { id firstName...

GraphiQL

▶

Prettify

History

1 mutation createUser(\$input: UserInput!) {

2 createUser(input: \$input) {

3 id

4 username

5 email

6 phone

7 firstName

8 lastName

9 }

10 }

QUERY VARIABLES

1 {

2 "input": {

3 "username": "test",

4 "email": "test@test.cz",

5 "phone": "479332973",

6 "firstName": "David",

7 "lastName": "Test"

8 }

9 }

{

{

"data": {

"createUser": {

"id": "8e278903-ca1a-49d2-9ede-1f757502861b",

"username": "test",

"email": "test@test.cz",

"phone": "479332973",

"firstName": "David",

"lastName": "Test"

}

}

}

< Mutation

User

×

Q Search User...

User type definition

FIELDS

id: ID!

username: String!

email: String

phone: String

firstName: String

lastName: String

Type System

GraphQL Types

```
type Query {  
  users: [User]  
  user(id: Int!): User  
}
```

```
type Task {  
  id: Int!  
  title: String  
  status: String  
  created_at: String  
}
```

```
type User {  
  id: Int!  
  name: String  
  created_at: String  
  tasks: [Task]  
}
```

Resolvers

Resolvers

```
{
  Query: {
    users: function(root, args, context, info){
      return new User()
        .fetchAll()
        .then(function(users){
          var nodes = users.models.map(function(user){
            return {
              id: this.user.get('id'),
              name: this.user.get('name'),
              created_at: this.user.get('created_at'),
            }
          });
          return nodes;
        });
    }
  }
}
```

Mutations

Mutations

creating a user

```
input userInput {  
  name: String  
}  
  
type Mutation {  
  createUser (input: userInput): User  
}
```

```
{  
  Mutation: {  
  
    // create a new user  
    createUser(root, { input }) {  
  
      return new User({ name: input.name }).save().then(function(user) {  
        var node = new UserTransformer(user).transform();  
        return node;  
      });  
    },  
  },  
}
```

Mutations

updating a user

```
type Mutation {  
  updateUser (id: Int!, input: userInput): User  
}
```

```
{  
  Mutation: {  
    updateUser(parent, { id, input }) {  
      return new User({ id: id }).fetch().then(function(user){  
        return user.save(input, { patch: true }).then(function(user) {  
          var node = new UserTransformer(user).transform();  
          return node;  
        });  
      })  
    },  
  },  
}
```


Validations

Validations

```
{
  Mutation: {
    // create a new user
    createUser(parent, { input }) {
      let validation = new Validator(input, {
        name: 'required|min:3'
      });

      if(validation.fails()) throw new ValidationException(validation.errors);

      return new User({ name: input.name }).save().then(function(user) {
        var node = new UserTransformer(user).transform();
        return node;
      });
    },
  },
}
```

Formatting errors

```
routes.use('/graphql', bodyParser.json(), graphqlExpress({
  schema,
  formatError: function(error){
    return {
      message: _.get(error, 'message', 'Something went wrong'),
      code: _.get(error, 'originalError.code', 400),
      errors: _.get(error, 'originalError.errors', [])
    }
  },
  formatResponse: function(response, options){
    if("errors" in response){
      let errors = response.errors;
      response = _.omit(response, 'data');
      response = _.omit(response, 'errors');
      response = errors[0];
    }
    return response;
  }
}));
```

N+1 Problem

Dataloader

<https://github.com/facebook/dataloader>

GraphQL Tools



<https://github.com/apollographql/graphql-tools>

Thank you