



Introduction to GraphQL

Learn about GraphQL, how it works, and how to use it. Looking for documentation on how to build a GraphQL service? There are libraries to help you implement GraphQL in [many different languages](#). For an in-depth learning experience with practical tutorials, see [How to GraphQL](#). Check out the free online course, [Exploring GraphQL: A Query Language for APIs](#).

GraphQL is a query language for your API, and a server-side runtime for executing queries using a type system you define for your data. GraphQL isn't tied to any specific database or storage engine and is instead backed by your existing code and data.

A GraphQL service is created by defining types and fields on those types, then providing functions for each field on each type. For example, a GraphQL service that tells you who the logged in user is (`me`) as well as that user's name might look like this:

```
type Query {  
  me: User  
}  
  
type User {  
  id: ID  
  name: String  
}
```

Along with functions for each field on each type:

```
function Query_me(request) {  
  return request.auth.user;  
}
```

LEARN

Introduction

Queries and Mutations

[Fields](#)[Arguments](#)[Aliases](#)[Fragments](#)[Operation Name](#)[Variables](#)[Directives](#)[Mutations](#)[Inline Fragments](#)

Schemas and Types

[Type System](#)[Type Language](#)[Object Types and Fields](#)[Arguments](#)[The Query and Mutation Types](#)[Scalar Types](#)[Enumeration Types](#)[Lists and Non-Null](#)[Interfaces](#)[Union Types](#)[Input Types](#)

Validation

Execution

Introspection

BEST PRACTICES

Introduction

Thinking in Graphs

Serving over HTTP

```
function User_name(user) {  
  return user.getName();  
}
```

[Authorization](#)[Pagination](#)[Global Object Identification](#)[Caching](#)

After a GraphQL service is running (typically at a URL on a web service), it can receive GraphQL queries to validate and execute. The service first checks a query to ensure it only refers to the types and fields defined, and then runs the provided functions to produce a result.

For example, the query:

```
{  
  me {  
    name  
  }  
}
```

Could produce the following JSON result:

```
{  
  "me": {  
    "name": "Luke Skywalker"  
  }  
}
```

To learn more, click **Continue Reading**.

[Continue Reading →](#)

Queries and Mutations

[Learn](#)[Code](#)[Community](#)[& More](#)[Introduction to GraphQL](#)[GitHub](#)[@graphql](#)[News Blog](#)

[Best Practices](#)[Frequently Asked Questions](#)[Training Courses](#)[GraphQL Specification](#)[Libraries & Tools](#)[Services & Vendors](#)[Discord](#)[Stack Overflow](#)[Resources](#)[Events](#)[Landscape](#)[GraphQL Foundation](#)[Logo and Brand Guidelines](#)[Code of Conduct](#)[Contact Us](#)[Edit this page](#)

Copyright © 2021 The GraphQL Foundation. All rights reserved. The Linux Foundation has registered trademarks and uses trademarks. For a list of trademarks of The Linux Foundation, please see our [Trademark Usage](#) page. Linux is a registered trademark of Linus Torvalds. [Privacy Policy](#) and [Terms of Use](#).